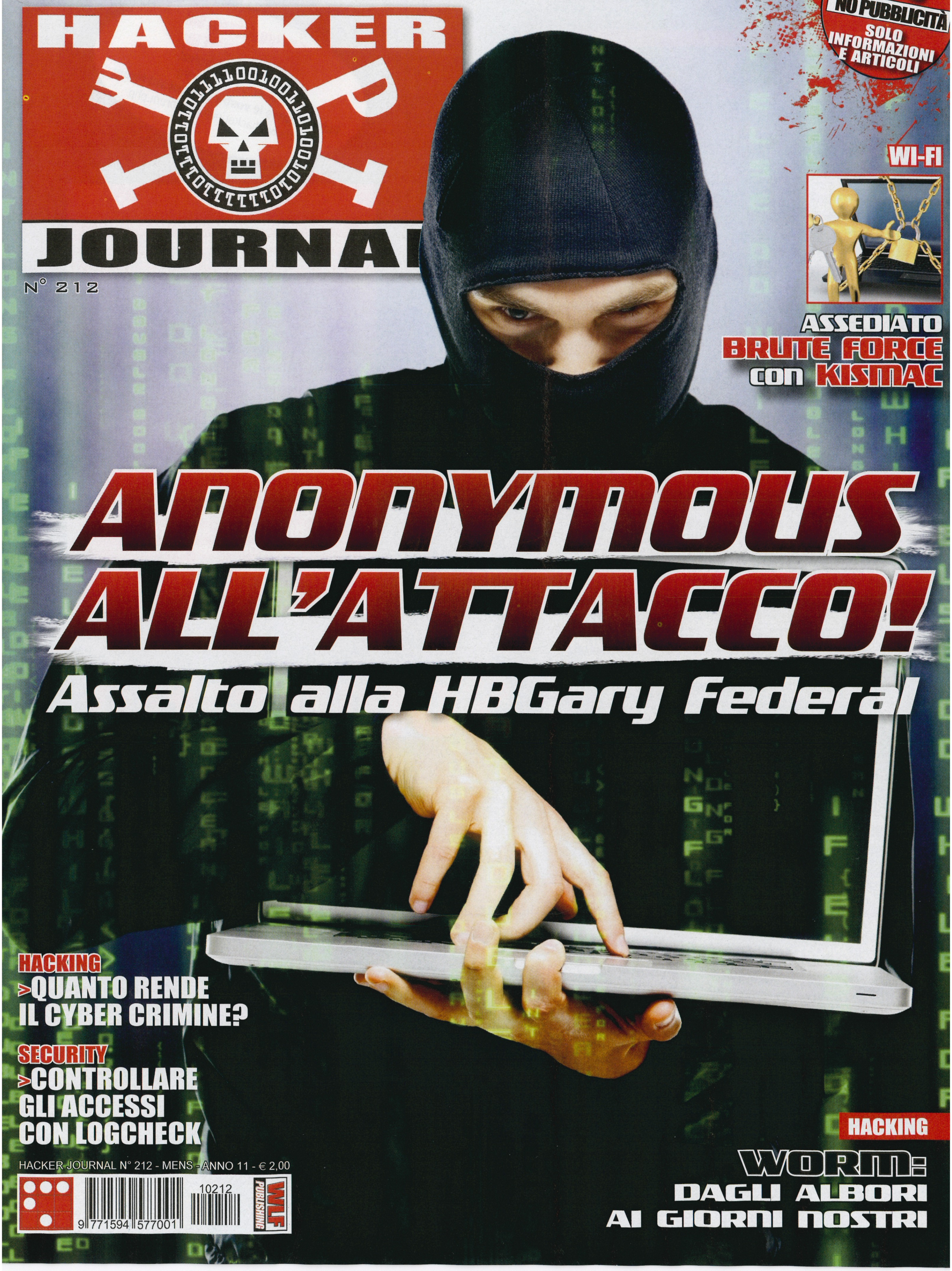
WWW.HACKERJOURNAL.IT

HACKER JOURNAL

N° 212

2€
NO PUBBLICITÀ
SOLO INFORMAZIONI E ARTICOLI



WI-FI

ASSEDIATO
BRUTE FORCE
con KISMAC

ANONYMOUS ALL'ATTACCO!

Assalto alla HBGary Federal

HACKING
>QUANTO RENDE IL CYBER CRIMINE?

SECURITY
>CONTROLLARE GLI ACCESSI CON LOGCHECK

HACKING
WORM: DAGLI ALBORI AI GIORNI NOSTRI

HACKER JOURNAL N° 212 - MENS - ANNO 11 - € 2,00
9 771594 577001 10212
WLF PUBLISHING

EDITORIALE

ANONYMOUS E ALTRO...

Si parla molto di Wikileaks, è normale, ma questo numero vuole puntare il focus sul gruppo di attivisti di Anonymous che si sono resi protagonisti di un attacco davvero clamoroso alla HBGary Federal, una società intimamente legata ai servizi di sicurezza nazionali e all'FBI.
È stato un attacco in punta di fioretto, condito con una grande ironia, che ricorda un po' le gesta degli eroi di cappa e spada, ma i cui effetti sono stati indubbiamente devastanti, almeno per la credibilità della società in questione e di chi la amministra.
Lo spunto di Anonymous ci consente di fare una piccola riflessione sull'importanza che ormai le tecniche di hackeraggio assumono in un contesto, quello globale, in cui le guerre, di qualsiasi entità, si conducono sempre più spesso con attacchi di tipo informatico. Ormai esistono vere e proprie task force di soldati "informatizzati" pronti a carpire informazioni o mettere fuori uso i servizi del nemico. Non importa che si tratti di terroristi o soldati di carriera. Questa è la nuova realtà con cui tutti dovranno presto o tardi fare i conti e che non mancherà di alimentare le cronache, incluse quelle di HJ. Chiuso il capitolo "cyber bellico", questo numero 212 ci regala un gioco di equilibrio godibile tra articoli di grande spessore tecnico e altri decisamente più discorsivi. Speriamo che gli uni e gli altri possano soddisfare le diverse aspettative di un pubblico, come quello di Hacker Journal, molto eterogeneo, costituito da lettori con una diversa preparazione tecnica ma accomunati da un unico comune denominatore: la curiosità.
Speriamo con questo numero 212 non tanto di creare nuove figure professionali, quanto di stimolare proprio quella voglia di ricercare, apprendere, capire che rappresenta un po' il volano di tutte le piccole grandi imprese.
Del resto come diceva il buon Eraclito: E' la curiosità che crea l'interesse, ed è la sazietà che lo distrugge...

RAGGIUNGETECI SUL NOSTRO CANALE IRC
Canale: #hackerjournal
Server: irc.azzurra.org
Fateci sapere le vostre opinioni sul forum
http://www.hackerjournal.it/forum.php

laboratorio@hackerjournal.it
Questo indirizzo è stato creato per inviare articoli, codici, spunti e idee. E' quindi proprio una sorta di "incubatore di idee".

posta@hackerjournal.it
E' l'account creato per l'omonima rubrica che è ricomparsa nelle pagine della rivista. A questo indirizzo dovete inviare tutte le mail che volete vengano pubblicate su HJ.

redazione@hackerjournal.it
Questo è l'indirizzo canonico. Quello con cui potete avere un filo diretto, sempre, con la redazione, per qualsiasi motivo che non rientri nelle due precedenti categorie di posta.

Sommario

ANNO 11 - N. 212
MARZO 2011

Editore: WLF Publishing S.r.l
Socio Unico medi & Son S.r.l
Via Torino 51 - 20063 Cernusco S/N (MI)
Tel. 02.924321 - Fax 02.92432236

Direttore responsabile: Teresa Carsaniga

Realizzazione Editoriale:
Progetti e Promozioni Srl
redazione@hackerjournal.it

Printing: Arti Grafiche Boccia Spa – 84131 Salerno

Distributore:
M-DIS Distribuzione Spa
Via Cazzaniga 19 – 20123 Milano

HACKER JOURNAL
Pubblicazione registrata al Tribunale di Milano il 27/10/03 con il numero 601

Una copia: euro 2,00

Cambia il mercato nero... del cyber crimine



Il mercato nero del cyber crimine, che inizialmente si focalizzava sulla diffusione di numeri di carte di credito e di credenziali di accesso a servizi bancari online rubati a utenti di tutto il mondo, sta diversificando il proprio modello d'affari offendo una vasta gamma di prodotti e servizi come password, finte carte di credito e molto altro. Nonostante queste informazioni siano facilmente reperibili, possono essere ottenute solo contattando direttamente gli hacker che ne promuovono la vendita su forum e chat.

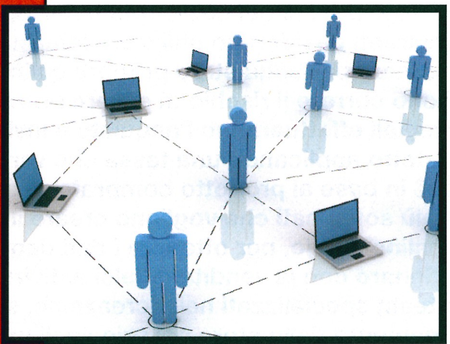
La vendita

Per avere accesso ai dati confidenziali, i criminali riescono facilmente a rubare l'account di un conto o di una carta di credito senza essere scoperti. Le informazioni di base ottenute possono poi essere acquistate a un prezzo non superiore ai 2$, senza garanzie sul saldo disponibile. Per avere maggiore sicurezza sulla disponibilità di denaro, il costo varia da 80$ per un fondo ridotto, fino a 700$ per avere accesso a un conto corrente con un saldo garantito di 82.000$. I prezzi possono aumentare se gli utenti derubati utilizzano il proprio conto per effettuare acquisti online oppure sfruttano piattaforme di pagamento come PayPal. In questo caso, per un conto semplice senza verifica del saldo il costo sarà di 10$, cifra che può arrivare fino a 1500$ in base alla piattaforma e ai fondi disponibili. I cyber criminali vendono anche carte di credito clonate (con un prezzo di base di 180$), macchinari per clonare carte di credito (il prezzo va dai 200$ ai 1000$) e falsi sistemi di prelievo automatico (a partire da 3500$ a salire, in base al modello). Servizi aggiuntivi, come il riciclaggio di denaro sporco (bonifici bancari o assegni da incassare) sono disponibili con commissioni che vanno dal 10% al 40% del costo totale dell'operazione. Se gli acquirenti desiderano utilizzare informazioni bancarie rubate per acquistare prodotti online ma non vogliono correre il rischio di essere rintracciati, i cyber criminali effettueranno l'acquisto e invieranno loro il prodotto applicando una tassa che varia dai 30$ ai 300$ in base al prodotto comprato. Per i cyber criminali più sofisticati che vogliono creare il proprio negozio online fittizio, per ottenere i dati degli utenti e rubare denaro con la vendita di falsi antivirus, esistono team specializzati nella creazione, sviluppo e posizionamento dello store nei principali motori di ricerca con un costo che varia a seconda del progetto. I prezzi per affittare botnet dalle quali inviare spam (ad esempio, attraverso computer zombie) mutano a seconda del numero di PC utilizzati, della frequenza di spam e della durata dell'affitto, da un minimo di 15$ fino a 20$ per l'utilizzo di un server SMTP o VPN che garantisca l'anonimato.

Il giro di affari

Il cosiddetto mercato nero del cyber crimine sfrutta le necessità degli acquirenti e funziona come un mercato tradizionale. Anche in questo tipo di commercio la concorrenza è spietata e la legge della domanda e dell'offerta obbliga i cyber criminali ad adeguare i prezzi e offrire sconti in base al volume d'acquisto. Molti offrono una prova d'accesso gratuita a conti correnti bancari o carte di credito rubate insieme alla formula "soddisfatti o rimborsati". Se il cliente non dovesse essere contento dell'acquisto il denaro speso gli verrà restituito e, se qualcosa non dovesse funzionare nei dati forniti, il "rivenditore" provvederà a sostituirli. Ovviamente, trattandosi di mercato nero, ci sono molti aspetti che differiscono completamente dal commercio tradizionale. Considerato che l'anonimato è la base di questo tipo di vendita, molti utilizzano forum nascosti per evitare di avere ospiti indesiderati. In questo modo, Internet è - a tutti gli effetti - il loro "ufficio" e nelle inserzioni pubblicitarie inseriscono anche gli orari di apertura al pubblico. I più audaci hanno account su Facebook e Twitter che utilizzano come vetrina per i loro prodotti e, per mantenere l'anonimato, il contatto avviene tramite applicazioni di messaggeria istantanea o account e-mail generici. Stabilito il contatto, la transazione può avvenire direttamente o attraverso l'accesso, con user e password - a un sito Internet creato dai rivenditori che permette di personalizzare il proprio carrello degli acquisti, proprio come avviene in ogni store online tradizionale e il pagamento viene quasi sempre effettuato in anticipo utilizzando servizi come Western Union, Liberty Riserve o WebMoney.

SOCIAL NETWORK: UNA POTENZIALE TRAPPOLA



Da una ricerca condotta da Microsoft in 11 paesi europei sul portale MSN, che ogni giorno raggiunge oltre 4 milioni di utenti in Italia, presentata in occasione del Safer Internet Day 2011 (www.saferinternet.it), il 40% dei teenager italiani sceglie autonomamente quali limiti all'accesso del proprio profilo adottare e il 15% sceglie di non adottarne alcuno. Eppure, in Italia il 60% dei minori dichiara di essere stato contattato almeno una volta da utenti sconosciuti. Se è vero che il 48% ha bloccato la richiesta di contatto, il 37% l'ha accettata per curiosità e il 7% ha deciso di non informare i propri genitori. Un dato allarmante è che il 66% decide di non rispondere ai messaggi inviati da gente estranea solo per disinteresse e non per prudenza. Altrettanto interessante è analizzare le motivazioni che portano i teenager sui social network. Il 43%, infatti, li utilizza per sfogarsi esprimendo idee e pensieri in maniera a volte anche aggressiva e il 25% dichiara di essere in cerca di amicizie con persone più adulte. Nonostante queste evidenze, il 32% dei genitori italiani non controlla in nessun modo il comportamento in Rete dei propri figli, anche se l'88% ha affrontato con loro l'argomento dei potenziali rischi del Web. Troppo spesso l'attenzione dei genitori è concentrata nell'impedire l'accesso a contenuti per gli adulti, come fa il 60% degli italiani, delegando invece nel 68% dei casi altre forme di precauzione direttamente ai propri figli.

AUMENTANO GLI ARRESTI DI CYBER CRIMINALI

Il 2010 è stato l'anno che ha fatto registrare il maggior numero di arresti e condanne di persone che hanno commesso crimini online. L'FBI ha reso noto di aver arrestato più di 90 persone, sospettate appartenere a una rete internazionale di criminali informatici e accusate di aver rubato circa 70 milioni di dollari da conti bancari negli Stati Uniti. Altri arresti sono stati effettuati nel Regno Unito e in Ucraina, da dove è stata diretta l'intera operazione. I criminali hanno ottenuto l'accesso ai dati di banking online inviando messaggi spam infetti. Secondo l'FBI, gli arresti rientrano in "uno dei più grandi casi di crimine informatico su cui si sia mai indagato". Un caso interessante che coinvolge gli strumenti di spionaggio installati su dispositivi mobili è stato reso noto da The Register lo scorso luglio. Le autorità rumene hanno arrestato 50 persone accusate di aver utilizzato software per monitorare le comunicazioni via telefono cellulare dei propri coniugi, concorrenti e altre persone. Sempre secondo The Register, l'Ente Rumeno per le Indagini sul Crimine Organizzato e il Terrorismo ha arrestato Dan Nicolae Oproiu, un esperto informatico di 30 anni, che avrebbe venduto spyware per dispositivi basati su sistemi operativi iPhone, Blackberry, Symbian e Windows Mobile.



AGGIORNARE E' MEGLIO CHE PREVENIRE

Windows XP continua a essere l'obiettivo principale degli hacker. Il sistema operativo Windows 7 è considerato più sicuro del suo predecessore Windows Vista. Tuttavia, nonostante abbia sorpassato Vista in termini di quota di mercato nel corso dell'anno, Windows 7 è ancora molto indietro rispetto a Windows XP, che resta di gran lunga il sistema operativo più popolare, nonché il principale obiettivo per molti creatori di malware. Utilizzare sistemi operativi datati comporta grossi rischi per la sicurezza. Ciò è stato dimostrato anche da rapporti che hanno individuato nei guasti ai computer ancora basati su sistema operativo Windows NT 4 del 1996, una delle possibili cause della fuoriuscita di petrolio nel Golfo del Messico.

TROJAN PER WINDOWS MOBILE

Una versione trojan del gioco di azione in 3D Anti-terrorist per Windows Mobile è stata caricata su numerosi siti Windows Mobile dai quali è possibile scaricare software gratuito. I telefoni infettati effettuavano telefonate verso numeri a pagamento particolarmente costosi, che risultavano in conti salati per le vittime.

DUE NUOVI CODICI COMPARSI IN TRE GIORNI PER ADESCARE GLI UTENTI DI FACEBOOK

Sono stati individuati due nuovi codici che utilizzano Facebook per adescare vittime ignare.
Il primo, Asprox.N, è un Trojan che raggiunge le sue potenziali vittime via email. Agli utenti arriva un messaggio che li avvisa che il loro account di Facebook è stato utilizzato per inviare spam e che, per ragioni di sicurezza, sono state modificate le credenziali di accesso. Il virus invia inoltre un finto file Word contenente la nuova password.
L'allegato ha un'insolita icona Word denominata Facebook_details.exe. Questo file è il vero e proprio Trojan che, una volta eseguito, scarica un file .doc che apre Word per far credere agli utenti che il programma originale sia stato aperto.
Inoltre, il Trojan scarica un altro file creato per aprire tutte le porte disponibili, connettendosi a diversi mail service provider per inviare spam al numero più elevato di utenti.
L'altro codice, Lolbot.Q, si diffonde attraverso applicazioni di messaggeria istantanea come MSN e Yahoo! con un messaggio che contiene un link pericoloso. Se aperto, scarica automaticamente un worm progettato per appropriarsi degli account di Facebook e impedirne l'utilizzo. Se l'utente tenta di accedere al proprio profilo, apparirà un messaggio che lo informa della disattivazione e della necessità di compilare un questionario per ripristinarlo, con la possibilità di vincere dei premi (come laptop o iPad) per incoraggiare a rispondere.
Dopo alcune domande, viene richiesto di inserire il numero di cellulare per ricevere dati per il download a un costo di €8.52 a settimana. Sottoscrivendo il servizio si riceve una password per poter riattivare l'accesso al proprio account.

di Massimiliano Rinaldi
redazione@hackerjournal.it

CATTIVISSIMO ME

Anonymous Vs HBGary Federal, uno degli attacchi più eclatanti di questo ultimo periodo in tutti i suoi particolari.

Chi sono i cattivi e chi i buoni? Difficile da dirsi, il titolo ispirato al film della Universal calza alla perfezione a questa storia di mosse e contromosse informatiche che, per il momento, ha visto un unico grande perdente Aaron Barr, amministratore delegato della HBGary Federal, un'agenzia creata per fornire consulenze sulla sicurezza informatica mirate alle necessità del dipartimento della Difesa, all'Fbi, alla Marina e alle altre agenzie del governo Usa. Ma procediamo con ordine…

ANONYMOUS VS HBGARY FEDERAL

L' HBGary Federal è un'agenzia che dovrebbe fare della sicurezza informatica il suo vessillo, offre consulenze ad altissimo livello, immaginiamo ben remunerate, proprio in questo settore, e, teoricamente, dovrebbe incarnare il ruolo dei buoni. Dall'altra parte si contrappongono gli hacker-attivisti di Anonymous, un gruppo che supporta la causa di WikiLeaks di cui ci siamo ampiamente occupati nel numero 210. Tecnicamente sarebbero i cattivi… La vicenda prende spunto da una serie di dichiarazioni di Aaron Barr secondo le quali la sua agenzia avrebbe voluto fornire i dati dei leader di Anonymous all'FBI, molto probabilmente in cambio di denaro. Molto. Insomma la prima mossa parte proprio da un attacco frontale di questa agenzia che, evidentemente, pensava di trarne un grosso profitto. Certo è che la risposta di Anonymous non si è fatta



attendere molto. Il gruppo hacker ha letteralmente defacciato il sito di HBGary Federal, inserendo come index una pagina, che vi proponiamo, in cui era disponibile anche un link da cui scaricare le oltre 50.000 e-mail private ricavate dall'attacco, circa 4GB di materiale. Materiale che in gran parte è stato cancellato dalla rete ma che è ancora disponibile all'indirizzo http://thepiratebay.org/torrent/6156166/HBGary_leaked_emails a cui, però, bisogna accedere attraverso un proxy in quanto il link di Pirate Bay è stato censurato in Italia e quindi reso irraggiungibile.
Nel documento di "defacement" viene rimarcata da Anonymous la loro natura di gruppo senza scopo di lucro che, proprio per questo, non può che deprecare e perseguire chiunque tenti di arricchirsi sfruttando il loro nome, le loro azioni e finalità. Insomma sembra che i ruoli della vicenda si scambino, gli hacker diventano virtuosi e l'agenzia di Governo un'entità dedita al solo arricchimento con qualsiasi mezzo. Nello stesso documento di defacciamento viene riportata una gustosa metafora: *Ti sei infilato ciecamente nell'alveare di Anonymous, un alveare dal quale hai provato a rubare il miele. Pensavi che le api non l'avrebbero difeso? Bene, eccoci qua. Hai fatto arrabbiare l'alveare, e ora stai per essere punto.*

IL COMUNICATO

Dopo avere messo a segno l'attacco è stato rilasciato Anonymous un comunicato stampa ufficiale, reperibile in lingua originale su anonnews.org all'indirizzo http://anonnews.org/?p=press&a=item&i=378. La traduzione in italiano suona più o meno così:
La ricerca di HBGary Federal è inaccurata, e non fornisce nessuna prova incriminante contro le persone nominate al suo interno. La mancanza di qualità nei dati raccolti ne annulla l'utilità. Barr ha dimenticato d'inserire una grande quantità d'informazioni disponibili online, infatti, ha perfino fallito nell'identificare i dati di persone che non avevano nessuna intenzione di nascondersi, come il Blogger di DailyKos Barrett Brown e l'amministratore di Anonnews.org (il sito dove è apparso il comunicato). Identità facilmente rintracciabili con una ricerca di qualche minuto su Google. Si tratta di una mossa assolutamente astuta e perfida. Fin dall'antichità non c'era peggiore onta per il nemico che quella di essere deriso e umiliato in combattimento: meglio la morte. Certo, erano altri tempi… In questo senso l'immagine di Aaron Barr (e della sua agenzia)

non ne esce certo bene. Al di là comunque delle parole, l'eclatante attacco alla HBGary Federal ha portato anche dei risvolti pratici: i membri di Anonymous sono entrati in possesso delle password di Twitter di Aaron Barr e se ne sono temporaneamente appropriati per diffondere il messaggio. Tramite il sito, hanno pubblicato tutti i suoi dati riservati, come indirizzi, numeri di previdenza sociale e sanitaria, e molto altro. Hanno inoltre avvisato che entro un'ora al proprietario sarebbe tornato il proprio account Twitter, in caso avesse "ammesso la sconfitta".

DOCUMENTAZIONE CONTROVERSA

Molti si sono soffermati solo sull'attacco di facciata che è stato ritenuto alla stregua di una semplice dimostrazione, e forse è stato proprio così, ma i dati recuperati dai messaggi delle mail sottratte non sono poi così irrilevanti, ne verrebbe fuori una strategia, a quanto pare mai realizzata, per difendere Bank of America e la Camera di commercio Usa dalle annunciate rivelazioni di WikiLeaks attraverso lo screditamento di quest'ultima. Infatti, HBGary Federal, in collaborazione con altre due compagnia, proponeva di far trapelare documenti falsi per poi contestare il sito di Assange. Un'altra strategia suggerita era di minacciare danni alle carriere dei giornalisti più strettamente legati a WikiLeaks. Nel frattempo non si fermano neanche le accuse nei confronti della HBGary che, secondo molti giornalisti indipendenti e blogger, starebbe sviluppando un rootkit in grado di rubare password o altre informazioni non rilevabile e, ipotesi questa ancora più interessante, avrebbe realizzato una copia del famigerato worm Stuxnet e, secondo quanto denuncia il sito CrowdLeaks, vorrebbe utilizzarlo per i propri fini invece che cercare di renderlo innocuo. Continua così la battaglia tra presunti buoni e cattivi accertati, in un'escalation di cui vi daremo informazione nei prossimi numeri e in cui niente e davvero così come sembra…

This domain has been seized by Anonymous under section #14 of the rules of the Internet.

Greetings HBGary (a computer "security" company),

Your recent claims of "infiltrating" Anonymous amuse us, and so do your attempts at using Anonymous as a means to garner press attention for yourself. How's this for attention?

You brought this upon yourself. You've tried to bite at the Anonymous hand, and now the Anonymous hand is bitch-slapping you in the face. You expected a counter-attack in the form of a verbal braul (as you so eloquently put it in one of your private emails), but now you've received the full fury of Anonymous. We award you no points.

What you seem to have failed to realize is that, just because you have the title and general appearence of a "security" company, you're nothing compared to Anonymous. You have little to no security knowledge. Your business thrives off charging ridiclous prices for simple things like NMAPs, and you don't deserve praise or even recognition as security experts. And now you turn to Anonymous for fame and attention? You're a pathetic gathering of media-whoring money-grabbing sycophants who want to reel in business for your equally pathetic company.

Let us teach you a lesson you'll never forget: you don't mess with Anonymous. You especially don't mess with Anonymous simply because you want to jump on a trend for public attention, which Aaron Barr admitted to in the following email:

"But its not about them...its about our audience having the right impression of our capability and the competency of our research. Anonymous will do what every they can to discredit that. and they have the mic so to speak because they are on Al Jazeera, ABC, CNN, etc. I am going to keep up the debate because I think it is good business but I will be smart about my public responses."

You've clearly overlooked something very obvious here: we are everyone and we are no one. If you swing a sword of malice into Anonymous' innards, we will simply engulf it. You cannot break us, you cannot harm us, even though you have clearly tried...

You think you've gathered full names and home addresses of the "higher-ups" of Anonymous? You haven't. You think Anonymous has a founder and various co-founders? False. You believe that you can sell the information you've found to the FBI? False. Now, why is this one false? We've seen your internal documents, all of them, and do you know what we did? We laughed. Most of the information you've "extracted" is publicly available via our IRC networks. The personal details of Anonymous "members" you think you've acquired are, quite simply, nonsense.

So why can't you sell this information to the FBI like you intended? Because we're going to give it to them for free. Your gloriously fallacious work can be a wonder for all to scour, as will all of your private emails (more than 66,000 beauties for the public to enjoy). Now as you're probably aware, Anonymous is quite serious when it comes to things like this, and usually we can elaborate gratuitously on our reasoning behind operations, but we will give you a simple explanation, because you seem like primitive people:

You have blindly charged into the Anonymous hive, a hive from which you've tried to steal honey. Did you think the bees would not defend it? Well here we are. You've angered the hive, and now you are being stung.

It would appear that security experts are not expertly secured.

```
We are Anonymous.
We are legion.
We do not forgive.
We do not forget.
Expect us - always.
```

Download HBGary email leaks

di G. Nervo
redazione@hackerjournal.it

LA PROGRAMMAZIONE FUNZIONALE



LINGUAGGI
DIFFERENZE, PREGI E ASPETTI DELLA PROGRAMMAZIONE FUNZIONALE RISPETTO A QUELLA IMPERATIVA.

La programmazione funzionale (functional programming) è un paradigma di programmazione nato verso la fine degli anni cinquanta in ambiti accademici che getta le sue radici nel lambda calcolo e nel concetto di funzione matematica. Per centinaia di anni, infatti, le funzioni hanno avuto un ruolo chiave nella matematica. Esse esprimono la connessione tra i parametri (input) ed il risultato (output). La principale differenza tra questo tipo di funzioni e le funzioni della programmazione imperativa è che le seconde possono avere degli effetti collaterali (side effects) perché possono cambiare risultato anche a seconda del valore delle variabili globali, esterne o non ricevute come paremetri, che utilizzano durante la computazione (getchar() in C, ricevendo gli stessi argomenti cambia output a seconda del contenuto del file descrittore stdin), mentre le prime restituiscono il loro output unicamente in base al loro input e le variabili presenti al loro interno sono le cosidette variabili matematiche, ossia degli identificatori che si riferiscono a dei valori immutabili e, quindi, non modificabili durante l'esecuzione della funzione (per esempio due chiamate di sqrt con lo stesso numero come input ritornerà sempre lo stesso valore).

LA FUNZIONE

In generale, possiamo dire che la programmazione funzionale si basa sul concetto di funzione (di solito anonima), che deve essere pura (pure function), di prima classe (first-class function) e di ordine superiore (higher-order function). Adesso vedrò di spiegarmi. Una funzione pura è quello che abbiamo descritto poco più sopra e credo non ci sia bisogno di dilungarsi oltre. Una funzione, invece, viene definita di prima classe quando restituisce un valore di output generato da un'altra funzione ricevuta come parametro (un esempio è la funzione map, che prende in input una funzione e una o più liste e ritorna il risultato della funzione applicata ad ogni elemento delle liste). Le funzioni di prim'ordine sono, invece, quelle che ricevono come parametri delle funzioni e producono come output altre funzioni (un esempio è una funzione adder, che riceve un numero e ritorna una funzione che a sua volta riceve un numero ritornando poi la somma del suo argomento con l'argomento della funzione che l'ha generata).

LA RICORSIONE

Un altro concetto strettamente legato alla programmazione funzionale, anche se ovviamente è presente in altri linguaggi, è la ricorsione. La ricorsione, come anche il più nabbo dei programmatori sa, è una tecnica che consiste nell'assegnare alle funzioni un caso base secondo il quale ritornare un determinato valore, e fare in modo che se il caso base risulti falso, la funzione richiami se stessa modificando uno o più parametri. La funzione ricorsiva, che di solito viene insegnata

per prima, è quella per calcolare il fattoriale di un numero (n!). Il caso base di questa funzione è verificare che il numero equivalga a zero e in tal caso ritornare 1 (se n equivale a zero, ritorna 1...), mentre la chiamata ricorsiva equivale a ri-chiamare n! in modo che n equivalga a (n – 1) (...altrimenti, ritorna n per il fattoriale di (n-1)). Esistono poi diversi tipi di funzioni ricorsive, questa era un esempio di tail recursion. Un altro tipo di ricorsione è la tree recursion, caso in cui una funzione richiama due o più volte se stessa (ad esempio, i Fibonacci). Nella programmazione funzionale la ricorsione sostituisce l'iterazione, e, infatti, molti compilatori/interpreti per i linguaggi basati su questo paradigma tendono ad ottimizzare il codice, gestendo in maniera efficace lo Stack (ovviamente ci sono anche linguaggi ibridi, che inglobano anche un po' di concetti della programmazione imperativa, come, ad esempio, Scheme, che offre anche delle primitive forme di iterazione). Il tipo di dato fodamentale di questo paradigma di programmazione è la lista collegata (linked list), una struttura di dati ricorsiva che al suo interno contiene due valori: l'elemento della lista e un puntatore che fa riferimento all'elemento successivo (un'altra linked list). Se la lista corrente è l'ultimo elemento il puntatore avrà un valore nullo (nil). Tutto ciò che non è una lista nella programmazione funzionale viene considerato un atomo (atom), così i numeri, i caratteri, ecc... sono atomi. Le liste poi possono essere scritte sotto forma di list comprehension (listcomp), un costrutto sintattico che viene usato per creare delle liste sul modello di liste pre-esistenti. In Haskell, la scrittura [(x * x) | x ← [1..10]], ad esempio, serve per creare una lista contenente tutti i quadrati dei numeri da 1 a 10.

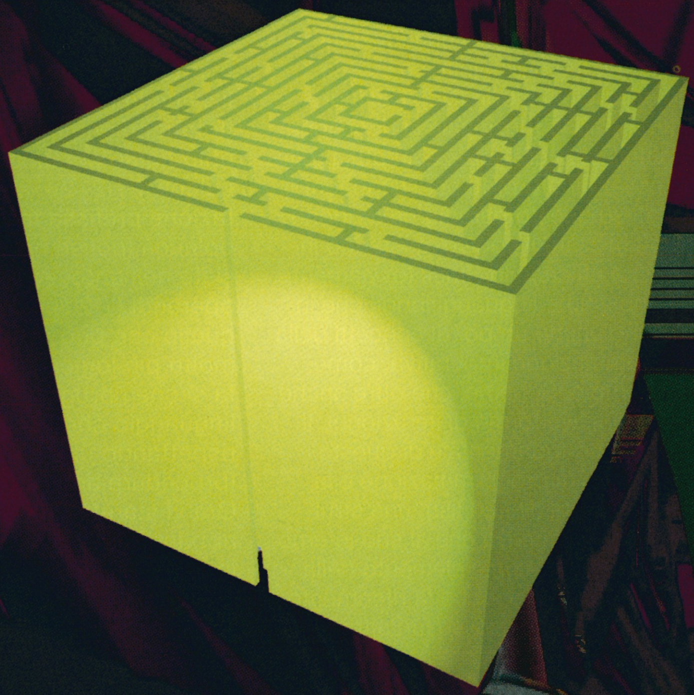
STRATEGIA DI VALUTAZIONE

Un altro punto cruciale della programmazione funzionale è la strategia di valutazione (evaluation strategy). Qui, i linguaggi funzionali si dividono in due. Da una parte, ci sono i linguaggi ibridi, come Scheme, che adattono la valutazione stretta (strict evaluation), in cui il risultato di una computazione viene calcolato all'istante, dall'altra parte i linguaggi più puri, come Haskell, che scelgono la valutazione pigra (lazy evaluation), in cui le espressioni vengono calcolate solo quando richieste. Inoltre, i linguaggi funzionali sono divisi anche sul tipo di notazione da usare per valutare le espressioni. Alcuni linguaggi utilizzano la notazione infissa (infix notation), che è la notazione che utilizziamo comunemente per fare i calcoli (2+(2+2)), altri usano la notazione prefissa (prefix notation), o polacca (polish notation), che consiste nell'antecedere l'operatore agli operandi (+ 2 (+ 2 2)). A dispetto di quello che può sembrare io trovo molto più logica la seconda, perché fa sì che i programmi scritti in un linguaggio che la usa si strutturino rispettando una semplice regola: (procedure arg1 arg2 … argN). Ora vi starete chiedendo perché abbandonare la programmazione imperativa per abbracciare quella funzionale. Beh, perché ve lo dico io... =D Ovviamente scherzo, anzi, non vi sto chiedendo nulla di tutto ciò. Io sono uno che generalmente per programmare usa lo stile imperativo/ad oggetti, ma lo stesso sono rimasto alquanto affascinato nello studiare questo paradigma, di colpo tutto mi è sembrato più coerente e scrivere codice in quello stile, beh... è fico. E' una delle tante esprienze di vita (di un programmatore) che devono essere provate e che di sicuro male non fanno. In base a queste esperienze poi ognuno imbocca la sua strada. Io, per esempio, ora uso Python, perché, oltre ad essere semplicemente unico, supporta anche alcuni concetti della programmazione funzionale, come le funzioni anonime lambda, le funzioni di ordine superiore e di prima classe, le listcomp e map, filter e reduce. Per fare il programmatore funzionale serio, invece, uso Scheme, che a mio avviso è l'ideale per chi vuole avvicinarsi a questo paradigma, senza dover abbandonare definitivamente quello imperativo. Poi c'è sempre Haskell, okay, ma questo è un altro discorso. Sinceramente, spero di non avervi annoiato più di tanto e, anzi, spero di aver stimolato in voi la curiosità che caratterizza un Hacker. D'altronde non posso spiegare un paradigma di programmazione con più di sessant'anni di vita in un solo articolo, supererei il limite di trenta pagine. Ma non temete, Google è vostro amico.



di Francesco Merlo
redazione@hackerjournal.it

ALGORITMI A BLOCCHI E ERRORI



CRITTOGRAFIA
COME RECUPERARE UN TESTO CRITTOGRAFATO (O ANCORA MEGLIO RIVELARLO) SFRUTTANDO LA PROPAGAZIONE DEGLI ERRORI CHE SI MANIFESTANO TRAI BLOCCHI DATI.

Tutti i sistemi di crittografia più utilizzati si basano su algoritmi simmetrici (DES; AES; BLOWFISH, ecc.), essendo gli algoritmi asimmetrici (RSA) utilizzati esclusivamente per lo scambio delle chiavi. Gli algoritmi più noti operano su blocchi di dati di lunghezza fissa ed utilizzano diverse modalità operative, legate o meno alla concatenazione dei dati. Le pagine seguenti intendono spiegare come, indipendentemente dall' algoritmo utilizzato, si possa risalire alla modalità utilizzata sfruttando la propagazione tra i blocchi dell'errore di un bit del testo crittato in fase di decrittografia; non entreranno, invece, in merito alla maggiore o minore sicurezza delle modalità stesse. L'argomento, in effetti, è poco sviluppato sia sulla letteratura corrente, sia nei corsi univer-sitari. La tecnica descritta, inoltre, consente, in alcuni casi, di correggere l'errore e di recuperare il testo originale. Le modalità esaminate sono:

- ECB (Electronic CodeBook), senza concatenazione;
- CBC (Cipher Block Chaining) con concatenazione;
- CFB (Cipher FeedBack) con concatenazione;
- OFB (Output FeedBack) senza concatenazione

Le figure utilizzate sono tratte, per comodità, dalla versione italiana di Wikipedia e sono state opportunamente modificate per evidenziare la propagazione degli errori.

CHIAVE DI LETTURA

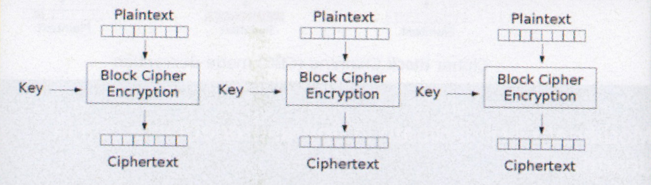
Simbolismo utilizzato. I simboli utilizzati sono i seguenti:
K= chiave simmetrica utilizzata;
PT= testo in chiaro;
B_{PT}= blocco di testo in chiaro;
CT= testo crittografato;
B_{CT}= blocco di testo crittografato;
IV= vettore di inizializzazione
PS= stringa pseudo casuale;
B_{PS}= blocco di stringa pseudocasuale;
$\oplus$= XOR[ii];
E_K(PT)= operazione di crittografia sotto la chiave K;
$D_K(_{CT})$= operazione di decrittografia sotto la chiave K;
$E_K(B_{PTn)}$= operazione di crittografia sotto la chiave K dell' ennesimo blocco di testo in chiaro;
$D_K(B_{CTn})$= operazione di decrittografia sotto la chiave K dell' ennesimo blocco di testo in chiaro;
II = blocchi affiancati.

Costruzione degli esempi:
Nell'analisi, per semplicità descrittiva, considereremo un blocco formato da otto bit (situa-zione che non si riscontra nella realtà). Nelle figure riportanti la propagazione dell'errore le caselle colorate in rosso sono quelle che contengono bit errati. Per evidenziare le differenze tra una modalità e l'altra, consideriamo sempre "errato" il bit 7 del secondo blocco (vuol dire che, in pratica, volendo utilizzare quanto descritto tale bit sarà portato da 1 a 0 o viceversa).

ECB

L'ECB (Electronic CodeBook) è il metodo base di utilizzo di un algoritmo a blocchi. Non vi è alcuna concatenazione tra gli stessi, Il testo da crittare viene suddiviso in n blocchi (nel nostro caso lunghi otto bit) ed ogni blocco viene crittografato/decrittogafato in maniera autonoma.

Quindi:
Operazione di crittografia



Electronic Codebook (ECB) mode encryption

Suddividiamo PT in n blocchi di 8 bit (nel caso l'ultimo blocco sia di lunghezza inferiore occorre usare una tecnica di "padding"):

$PT = B_{PT1} || B_{PT2} || B_{PT3} ||....|| B_{PTn}$

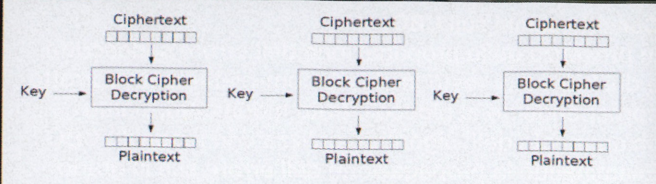
Crittografiamo ora ogni blocco sotto la chiave K:

$B_{CT1} = E_K(B_{PT1})$
$B_{CT2} = E_K(B_{PT2})$
$B_{CT3} = EK(B_{PT3})$
....
$B_{CTN} = E_K(B_{PTn})$

Il testo crittografato (trasmesso o archiviato) sarà::

$CT = B_{CT1} || B_{CT2} || B_{CT3} ||....|| B_{CTn}$

Operazione di decrittografia



Electronic Codebook (ECB) mode decryption

Suddividiamo CT in n blocchi di 8 bit (Avendo, eventualmente, usato una tecnica di "pad-ding" tutti i blocchi sono di lunghezza 8):

$CT = B_{CT1} || B_{CT2} || B_{CT3} ||....|| B_{CTn}$

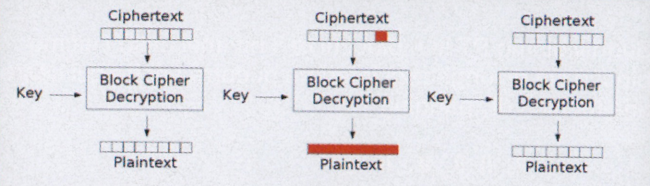
Decrittografiamo ora ogni blocco sotto la chiave K:

$B_{PT1} = D_K(B_{CT1})$
$B_{PT2} = D_K(B_{CT2})$
$B_{PT3} = D_K(B_{CT3})$
....
$B_{PTn} = D_K(B_{CTn})$

Il testo decrittografato sarà (nel caso l'ultimo blocco sia di lunghezza inferiore occorre eli-minare il "padding"):

$PT = B_{PT1} || B_{PT2} || B_{PT3} ||....|| B_{PTn}$

Propagazione dell'errore



Electronic Codebook (ECB) mode decryption

Modificato il bit indicato in figura, procediamo con la decrittografia:

$CT = B_{CT1} || BC_{T2} || B_{CT3} ||....|| B_{CTn}$

Decrittografiamo ora ogni blocco sotto la chiave K:

$B_{PT1} = D_K(B_{CT1})$
$BP_{T2} = D_K(B_{CT2})$ Blocco errato
$B_{PT3} = D_K(B_{CT3})$
....
$B_{PTn} = D_K(B_{CTn})$

Il testo decrittografato sarà (nel caso l'ultimo blocco sia di lunghezza inferiore occorre eli-minare il "padding"):

$PT = B_{PT1} ||$ Blocco $errato_2 || B_{PT3} ||....|| B_{PTn}$

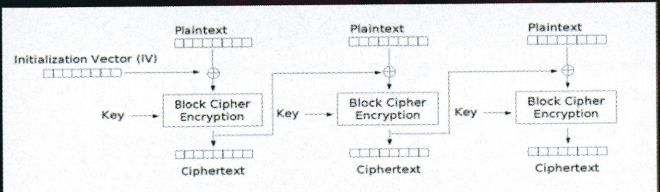
Come si nota dalla figura e dalla descrizione del processo di decrittografia, il blocco con-tente il bit errato verrà decrittato in modo del tutto errato, ma non vi sarà propagazione dell'errore nei blocchi successivi.

Considerazioni sull'ECB
In assenza di propagazione dell'errore, ed in presenza del blocco corrispondente di testo in chiaro completamente corrotto, siamo chiaramente in presenza dell'algoritmo utilizzato in modalità ECB.
Tale modalità è l'unica, tra quelle presentate, che non consente il recupero dell' errore.

CBC

Il CBC (Cipher Block Chaining) è, probabilmente, la modalità più utilizzata dagli algoritmi simmetrici, sia in fase di crittografia, sia in fase di autenticazione. Opera concatenando il testo in chiaro del blocco da crittografare, prima di sottoporlo alla crittografia stessa, con il testo crittografato del blocco precedente.

Operazione di crittografia



Cipher Block Chaining (CBC) mode encryption

Suddividiamo PT in n blocchi di 8 bit (nel caso l'ultimo blocco sia di lunghezza inferiore occorre anche qui usare una tecnica di "padding"):

$PT = B_{PT1} \| B_{PT2} \| B_{PT3} \| \ldots \| B_{PTn}$

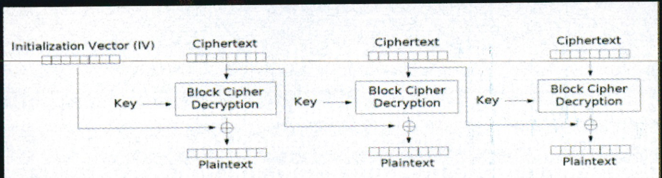
Crittografiamo ora ogni blocco concatenandolo con il crittografato del blocco precedente sotto la chiave K (nel caso del primo blocco la concatenazione viene fatta con il Vettore di Inizializzazione):

$$B_{CT1} = E_K(B_{PT1} \oplus IV)$$
$$B_{CT2} = E_K(B_{PT2} \oplus B_{CT1})$$
$$B_{CT3} = E_K(B_{PT3} \oplus B_{CT2})$$
....
$$B_{CTn} = EK(B_{PTn} \oplus B_{CTn-1})$$

Il testo crittografato (trasmesso o archiviato) sarà::

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

Operazione di decrittografia



Cipher Block Chaining (CBC) mode decryption

Suddividiamo CT in n blocchi di 8 bit (Avendo, eventualmente, usato una tecnica di "pad-ding" tutti i blocchi sono di lunghezza 8):

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

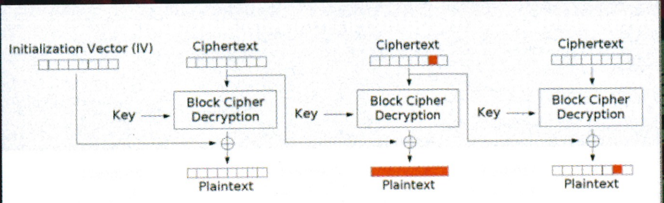
Decrittografiamo ora ogni blocco sotto la chiave K: e concateniamo il blocco ottenuto con il crittografato del blocco precedente (nel caso del primo blocco la concatenazione viene fat-ta con il Vettore di Inizializzazione):

$$B_{PT1} = D_K(B_{CT1}) \oplus IV$$
$$B_{PT2} = D_K(B_{CT2}) \oplus B_{CT1}$$
$$B_{PT3} = D_K(B_{CT3}) \oplus B_{CT2}$$
....
$$B_{PTn} = D_K(B_{CTn}) \oplus B_{CTn-1}$$

Il testo decrittografato (nel caso l'ultimo blocco sia di lunghezza inferiore occorre eliminare il "padding") sarà:

$PT = B_{PT1} \| B_{PT2} \| B_{PT3} \| \ldots \| B_{PTn}$

Propagazione dell'errore



Cipher Block Chaining (CBC) mode decryption

Modificato il bit indicato in figura, procediamo con la decrittografia:

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

Decrittografiamo ora ogni blocco sotto la chiave K:

$B_{PT1} = D_K(B_{CT1}) \oplus IV$
$B_{PT2} = D_K(B_{CT2}) \oplus B_{CT1}$ Blocco errato
$B_{PT3} = D_K(B_{CT3}) \oplus B_{CT2}$ Blocco con bit errato
....
$B_{PTn} = D_K(B_{CTn}) \oplus B_{CTn-1}$

Il testo decrittografato sarà (nel caso l'ultimo blocco sia di lunghezza inferiore occorre eli-minare il "padding"):

$PT = B_{PT1} \|$ Blocco errato$_2 \|$ Blocco con bit errato$_3 \| \ldots \| B_{PTn}$

Come si nota dalla figura e dalla descrizione del processo di decrittografia, il blocco con-tente il bit errato verrà decrittato in modo del tutto errato, e la funzione ⊕ propaga l'errore nel bit del blocco successivo corrispondente al bit.

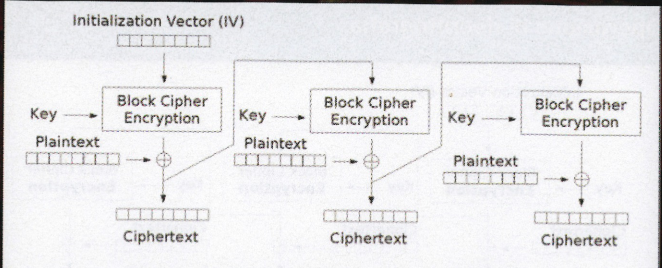
Considerazioni sull'CBC.

In presenza della propagazione dell'errore sudescritta, e del blocco di testo in chiaro com-pletamente corrotto corrispondente al blocco cifrato modificato, siamo chiaramente dell' algoritmo utilizzato in modalità CBC.. Esaminando la propagazione dell'errore si è in grado di individuare il bit (o i bit) modificato e ripristinare la situazione corretta.

CFB

La modalità CFB (Cipher FeedBack) opera concatenando il testo in chiaro del blocco da crittografare con il testo crittografato del blocco precedente.

Operazione di crittografia



Cipher Feedback (CFB) mode encryption

Suddividiamo PT in n blocchi di 8 bit (nel caso l'ultimo blocco sia di lunghezza inferiore non occorre, in questo caso, usare una tecnica di "padding", basta troncare l'ultimo blocco crittografato alla giusta lunghezza):

$PT = B_{PT1} \| B_{PT2} \| B_{PT3} \| \ldots \| B_{PTn}$

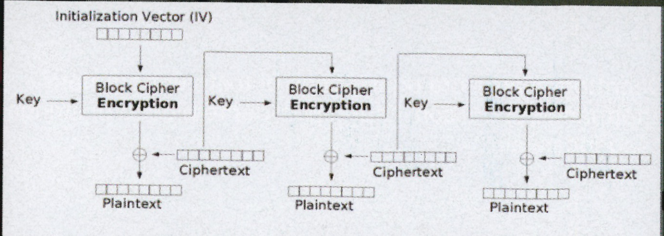
Crittografiamo ora ogni blocco concatenandolo con il crittografato del blocco precedente sotto la chiave K (nel caso del primo blocco la concatenazione viene fatta con il crittografa-to del Vettore di Inizializzazione):

$B_{CT1} = E_K(IV) \oplus B_{PT1}$
$B_{CT2} = E_K(B_{CT1}) \oplus B_{PT2}$
$B_{CT3} = E_K(B_{CT2}) \oplus B_{PT3}$
....
$B_{CTn} = E_K(B_{CTn-1}) \oplus BPTn$

Il testo crittografato (trasmesso o archiviato) sarà::

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

Operazione di decrittografia



Cipher Feedback (CFB) mode decryption

Il processo di decrittografia, in questa modalità, è uguale a quello di crittografia, quindi l'algoritmo non viene mai utilizzato in modalità decrittografica: l'unica differenza è che la funzione ⊕ viene utilizzata tra blocchi crittografati. Suddividiamo CT in n blocchi di 8 bit (Avendo, eventualmente, usato una tecnica di "pad-ding" tutti i blocchi sono di lunghezza 8):

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

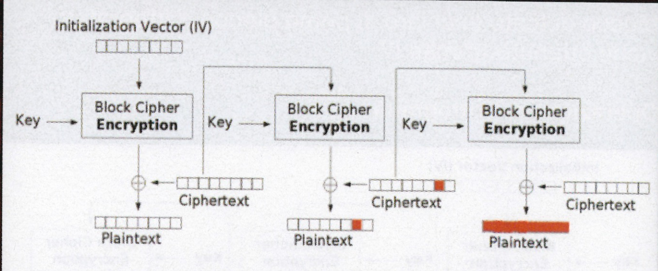
Crittografiamo ora ogni blocco sotto la chiave K: e concateniamo il blocco ottenuto con il crittografato del blocco stesso (nel caso del primo blocco la concatenazione viene fatta con il crittografato del Vettore di Inizializzazione):

$B_{PT1} = E_K(IV) \oplus B_{CT1}$
$B_{PT2} = E_K(B_{CT1}) \oplus B_{CT2}$
$B_{PT3} = E_K(B_{CT2}) \oplus B_{CT3}$
....
$B_{PTn} = E_K(B_{CTn-1}) \oplus B_{CTn}$

Il testo decrittografato sarà (nel caso l'ultimo blocco sia di lunghezza inferiore occorre eli-minare i bit in soprannumero):

$PT = B_{PT1} \| B_{PT2} \| B_{PT3} \| \ldots \| B_{PTn}$

Propagazione dell'errore



Cipher Feedback (CFB) mode decryption

Modificato il bit indicato in figura, procediamo con la decrittografia:

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

Crittografiamo ora ogni blocco sotto la chiave K: e concateniamo il blocco ottenuto con il crittografato del blocco stesso (nel caso del primo blocco la concatenazione viene fatta con il crittografato del Vettore di Inizializzazione):

$B_{PT1} = E_K(IV) \oplus B_{CT1}$
$B_{PT2} = E_K(B_{CT1}) \oplus B_{CT2}$ Blocco con bit errato
$BP_{T3} = E_K(B_{CT2}) \oplus B_{CT3}$ Blocco errato
....
$B_{PTn} = E_K(B_{CTn-1}) \oplus B_{CTn}$

Il testo decrittografato sarà (nel caso l'ultimo blocco sia di lunghezza inferiore occorre eli-minare i bit in soprannumero):

$PT = B_{PT1} \|$ Blocco con bit errato$_2$ $\|$ Blocco errato$_3$ $\| \ldots \| B_{PTn}$

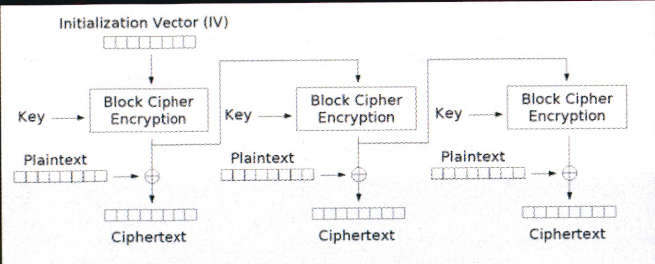
Come si nota dalla figura e dalla descrizione del processo di decrittografia, il blocco con-tente il bit errato verrà decrittato con il bit corrispondente corrotto, la funzione ⊕ propaga l'errore ed il blocco successivo verrà decrittato in modo del tutto errato.

Considerazioni sull'CFB.
In presenza della propagazione dell'errore sudescritta, e del blocco di testo in chiaro com-pletamente corrotto susseguente al blocco cifrato modificato, siamo chiaramente dell' al-goritmo utilizzato in modalità CFB.. Esaminando la propagazione dell'errore si è in grado di individuare il bit (o i bit) modificato e ripristinare la situazione corretta.

OFB

L'OFB (Output FeedBack) è la seconda modalità, tra quelle esaminate, che non opera concatenazioni tra testo in chiaro e testo crittografato. Il PT non viene mai sottoposto all'algoritmo di crittografia che opera come un generatore di stringa pseudo casuale, il ri-sultato del crittografato viene sottoposto alla funzione ⊕ con il blocco di testo in chiaro. Il primo blocco crittografato viene ottenuto tramite IV.

Operazione di crittografia



Output Feedback (OFB) mode encryption

Pur non essendo necessario, essendo una crittografia di flusso, suddividiamo per sempli-cità espositiva PT in n blocchi di 8 bit (nel caso l'ultimo blocco sia di lunghezza inferiore non occorre, in questo caso, usare una tecnica di "padding", basta troncare l'ultimo blocco crittografato alla giusta lunghezza):

$PT = B_{PT1} \| B_{PT2} \| B_{PT3} \| .... \| B_{PTn}$

Generiamo, in questo modo, l stringa pseudocasuale:

$B_{PS1} = E_K(IV)$
$B_{PS2} = E_K(B_{PS1})$
$B_{PS3} = E_K(B_{PS2})$
....
$B_{PSn} = E_K(B_{PSn-1})$

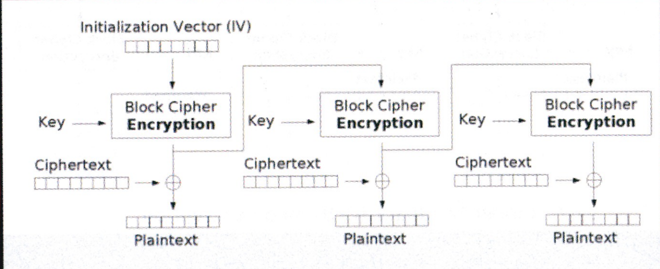
Crittografiamo ora ogni blocco di testo in chiaro effettuando uno ⊕ con il corrispondente blocco di stringa pseudocasuale

$B_{CT1} = B_{PS1} \oplus B_{PT1}$
$B_{CT2} = B_{PS2} \oplus B_{PT2}$
$B_{CT3} = B_{PS3} \oplus B_{PT3}$
....
$B_{CTn} = B_{PSn} \oplus B_{PTn}$

Il testo crittografato (trasmesso o archiviato) sarà::

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| .... \| B_{CTn}$

Operazione di decrittografia



Output Feedback (OFB) mode decryption

Il processo di decrittografia, in questa modalità, è uguale a quello di crittografia, quindi l'algoritmo non viene mai utilizzato in modalità decrittografica: l'unica differenza è che la funzione ⊕ viene utilizzata tra blocchi PS ed i corrispondenti blocchi crittografati.

Pur non essendo necessario, essendo una decrittografia di flusso, suddividiamo per sem-plicità espositiva CT in n blocchi di 8 bit (nel caso l'ultimo blocco sia di lunghezza inferiore non occorre, in questo caso, usare una tecnica di "padding", basta troncare l'ultimo blocco crittografato alla giusta lunghezza):

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| .... \| B_{CTn}$

Rigeneriamo la stringa pseudocasuale:

$B_{PS1} = E_K(IV)$
$B_{PS2} = E_K(B_{PS1})$
$B_{PS3} = E_K(B_{PS2})$
....
$B_{PSn} = E_K(_{BPSn-1})$

Decrittografiamo ora ogni blocco di testo crittografato effettuando uno ⊠ con il corrispon-dente blocco di stringa pseudocasuale

$B_{PT1} = B_{PS1} \oplus B_{CT1}$
$B_{PT2} = B_{PS2} \oplus B_{CT2}$
$B_{PT3} = B_{PS3} \oplus B_{CT3}$
....
$BPTn = B_{PSn} \oplus B_{CPTn}$

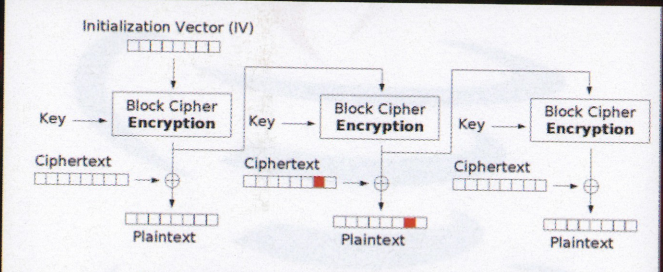
Il testo in chiaro sarà::

$PT = B_{PT1} \| B_{PT2} \| B_{PT3} \| .... \| B_{PTn}$



“Tutti i sistemi di crittografia più utilizzati si basano su algoritmi simmetrici (DES; AES; BLOWFISH, ecc.), essendo gli algoritmi asimmetrici (RSA) utilizzati esclusivamente per lo scambio delle chiavi”

Propagazione dell’errore



Output Feedback (OFB) mode decryption

Modificato il bit indicato in figura, procediamo con la decrittografia:

$CT = B_{CT1} \| B_{CT2} \| B_{CT3} \| \ldots \| B_{CTn}$

Rigeneriamo la stringa pseudocasuale:

$B_{PS1} = E_K(IV)$
$B_{PS2} = E_K(B_{PS1})$
$B_{PS3} = E_K(B_{PS2})$
....
$B_{PSn} = E_K(B_{PSn-1})$

Decrittografiamo ora ogni blocco di testo crittografato effettuando uno ⊕ con il corrispon-dente blocco di stringa pseudocasuale

$B_{PT1} = B_{PS1} \oplus B_{CT1}$
$B_{PT2} = B_{PS2} \oplus B_{CT2}$ Blocco con bit errato
$B_{PT3} = B_{PS3} \oplus B_{CT3}$
....
$B_{PTn} = B_{PSn} \oplus B_{CPTn}$

Il testo in chiaro sarà::

$PT = B_{PT1} \|$ Blocco con bit errato$_2$ $\| B_{PT3} \| \ldots \| B_{PTn}$

Come si nota dalla figura e dalla descrizione del processo di decrittografia, il blocco con-tente il bit errato verrà decrittato con il bit corrispondente corrotto, mentre non esiste pro-pagazione dell’errore.

Considerazioni sull’OFB

In presenza della corruzione del solo bit corrispondente al bit modificato nel testo crittogra-fato, senza altra propagazione, siamo chiaramente in presenza di una modalità OFB o di un sistema crittografico utilizzante non algortmi crittografici a blocchi, ma crittografia di flusso. Si è in grado, quindi, di individuare il bit (o i bit) modificato e ripristinare la situazio-ne corretta.

Note

Il metodo non funziona se il prodotto utilizzato verifica la correttezza del testo da decrittare, es. mediante un CRC: In tal caso non verrà effettuata la decrittografia e verrà segnalato il file come corrotto.

Funzione logica che opera nel seguente modo: 0 ⊠ 0 = 0, 0 ⊕ 1 = 1, 1 ⊕ 0 = 1, 1 ⊕ 1 = 0

Quindi:

110011
⊕
100111
______
010100

di Mauro Fossi
redazione@hackerjournal.it

UNA CLASSE "SERIALIZZATA"

SERIALIZZARE UN OGGETTO SIGNIFICA TRASFORMARLO IN UNA SEQUENZA DI BYTE ALLO SCOPO DI RICOSTRUIRLO IN UN MOMENTO SUCCESSIVO. VEDIAMO IN QUESTO ARTICOLO COME SIA POSSIBILE CON L'USO DELLE CLASSI IN JAVA GESTIRE, ESTRARRE ED UTILIZZARE I DATI IN AMBIENTI ETEROGENEI.

Il presente articolo vuole dimostrare come si possano utilizzare i meccanismi di serializzazione messi a disposizione dall'ambiente di programmazione Java per costruire, salvare su supporti eterogenei, trasmettere lungo un'infrastruttura di rete, oggetti che rappresentano determinate informazioni. A questo punto è importante fornire immediatamente, soprattutto per i meno esperti, la risposta a due domande che sorgono spontanee:

-che cosa intendiamo per "ambiente di programmazione Java";
-che cosa significa serializzazione.

Per quanto concerne il primo quesito, è importante chiarire come considerare Java un mero linguaggio di programmazione sia sicuramente riduttivo: Java, nelle sue forme di Standard ed Enterprise Edition rappresenta, infatti, un completo ambiente di sviluppo che mette a disposizione del programmatore un grande numero di classi e di librerie in grado di fornire un grande numero di funzionalità. Allo stesso tempo, il fatto che i nostri programmi abbiano bisogno di un particolare interprete (ovviamente la Virtual Machine) per essere eseguiti, rende più idonea la definizione di "ambiente" all'interno del quale progettare, compilare ed eseguire il nostro codice. Per quanto riguarda la seconda domanda, una risposta sintetica ma allo stesso tempo abbastanza completa, può essere la seguente: serializzare un oggetto significa trasformarlo in una sequenza di byte allo scopo di ricostruirlo in un momento successivo (dopo averne fatto l'uso desiderato, per esempio averlo salvato in un file, in un database, od averlo spedito lungo la rete). Il percorso logico, nella sua forma più semplice, può essere così sintetizzato:

1)si crea una classe per rappresentare una determinata

informazione;
2)si utilizzano i metodi e le proprietà della classe per immagazzinare i dati;
3)si salva come sequenza di byte;
4)successivamente si utilizza la classe per estrarre i dati in un formato più facilmente comprensibile.

La condizione necessaria è che sia chi scrive i dati, sia chi successivamente li estrarrà (non necessariamente con la stessa applicazione), devono conoscere come è definita la classe "contenitore". Di fatto, stiamo costruendo un dialogo tra due o più agenti che condividono una CONVENZIONE, allo scopo di parlare un linguaggio comune. Prenderemo in esame un esempio scritto in Java, ma nulla vieta (anzi lo potremo sperimentare in puntate successive) di costruire lo stesso sistema utilizzando altri linguaggi di programmazione. Allo scopo di far venire al lettore "l'acquolina in bocca", o quanto meno di stimolarne l'attenzione sulle potenzialmente infinite applicazioni di questa tecnica, possiamo elencare alcune applicazioni, frutto della mia esperienza lavorativa diretta:
-estrazione dei record da un database Mysql e spedizione dell'oggetto creato (e criptato) ad una macchina remota;
-spedizione ad un server remoto di un oggetto che contiene tre immagini Jpeg (small, medium, big) relative alla foto di un certo personaggio, insieme a dei valori stringa che rappresentano diverse informazioni sul personaggio ritratto (nome, professione, ecc.). L'oggetto viene spedito al server dell'agenzia fotografica che salva su disco le immagini e salva in un database le informazioni, allo scopo di fornire un motore di ricerca per il proprio archivio;
-salvataggio dei dati di un piccolo programma gestionale, evitando di dover appoggiarsi ad un database o di dover trafficare con troppi file di testo.

UNA "CLASSE" PER IL PUGILE

Inizieremo con un esempio abbastanza semplice: costruiremo una classe per memorizzare la foto ed alcune informazioni di un famoso atleta (un grande pugile) e la serializzeremo salvandola su un file. Successivamente, effettueremo l'operazione inversa: l'estrazione di tutti questi dati. Iniziamo a costruire la classe "contenitore" dei nostri dati:

```
import java.io.*;

class Pugile implements Serializable { //Abbiamo necessità di implementare l'interfaccia java.io.Serializable

private String nome;
private String cognome;
private String nickname;
private int anni;
private String categoria;
private String record;
private byte[] bImg; //Array di bytes che utilizzeremo per memorizzare la foto in formato .jpg

//Metodi per leggere e scrivere i dati relativi al pugile:

public String getNome() { return this.nome; }

public void setNome(String nome) { this.nome=nome; }

public String getCognome() { return this.cognome; }

public void setCognome(String cognome) { this.cognome=cognome; }

public String getNickname() { return this.nickname; }

public void setNickname(String nickname) { this.nickname=nickname; }

public int getAnni() { return this.anni; }

public void setAnni(int anni) { this.anni=anni; }

public String getCategoria() { return this.categoria; }

public void setCategoria(String categoria) { this.categoria=categoria; }

public String getRecord() { return this.record; }

public void setRecord(String record) { this.record=record; }

//Metodi per salvare ed estrarre i byte della foto .jpg

public byte[] getBImg() { return this.bImg; }

public void setBImg(byte[] bImg) { this.bImg=bImg; }

}
```

La classe è molto semplice. Rimarchiamo un piccolo particolare: le proprietà sono private, per accedervi in lettura e scrittura utilizzeremo pertanto i metodi (pubblici). In questo modo rispettiamo quello che viene definito il principio dell' "information hiding". Passiamo alle due piccole classi che utilizzeremo per salvare e successivamente estrarre il nostro oggetto. Io utilizzerò un'immagine "tito.jpg" presente sul mio hard-disk. Voi utilizzate quella che preferite. Per brevità valorizzeremo i dati del pugile (incluso il nome dell'immagine jpg) direttamente nel codice. Ovviamente in un'applicazione "reale" e non di esempio, questi verranno ricevuti dinamicamente. Potete provare, ad esempio, a modificare il codice passandoli da linea di comando come parametri del metodo main().

```
import java.io.*;

class Scrittore {

private void scrivi() {
 try {
  Pugile p=new Pugile();
  //Scrivo le informazioni:
  p.setNome("Felix");
  p.setCognome("Trinidad");
  p.setNickname("Tito");
  p.setAnni(32);
  p.setCategoria("Superwelter");
  p.setRecord("42-1-0"); //Vittorie-sconfitte-
pareggi ;-)
  //Leggo l'immagine, la salvo in un array di byte
e la passo al metodo setBImg()
  FileInputStream fis=new FileInputStream("tito.
jpg");
  ByteArrayOutputStream baos=new
ByteArrayOutputStream();
  int c;
   while((c=fis.read())!=-1) { baos.write(c); }
  p.setBImg(baos.toByteArray());
  fis.close();
  baos.close();
  //Utilizzeremo la classe ObjectOutputStream per
serializzare i dati e salvarli nel file tito.dat:
  ObjectOutputStream oos=new ObjectOutputStream(new
FileOutputStream("tito.dat"));
  oos.writeObject(p);
  oos.close();
 }
 catch(Exception e) {
  System.err.println("Si e' verificato l'errore:
"+e.toString());
 }
}

public static void main(String args[]) { new
Scrittore().scrivi(); }

}
```

Anche questa classe è piuttosto semplice. Utilizziamo il metodo writeObject() della classe ObjectOutputStream per serializzare e salvare il nostro oggetto. La gestione delle eccezioni è molto semplificata, ma il nostro è solo un primo esempio!

Passiamo adesso alla terza classe, che leggerà dal file tito.dat l'oggetto serializzato e "magicamente" ci restituirà i nostri dati, oltre a ricostruire l'immagine del pugile salvandola in un file tito2.jpg identico all'originale.

```
import java.io.*;

class Lettore {

private void leggi() {
 try {
  //Utilizzeremo la classe ObjectInputStream per
leggere l'oggetto serializzato dal file tito.dat:
  ObjectInputStream ois=new ObjectInputStream(new
FileInputStream("tito.dat"));
  Object obj=ois.readObject();
  ois.close();
  //E' necessario un casting, se obj non fosse
un'istanza della classe Pugile verrebbe generata
una ClassCastException:
  Pugile p=(Pugile)obj;
  //A questo punto possiamo estrarre i dati e
stamparli sullo standard output:
  System.out.println("Nome e cognome: "+p.
getNome()+" "+p.getCognome());
  System.out.println("Nickname: "+p.getNickname());
  System.out.println("Anni: "+String.valueOf(p.
getAnni()));
  System.out.println("Categoria: "+p.
getCategoria());
  System.out.println("Record: "+p.getRecord());
  //Non resta che estrarre i byte per ricostruire
l'immagine:
  byte bImg[]=p.getBImg();
  new FileOutputStream("tito_bis.jpg").write(bImg);
 }
 catch(Exception e) {
  System.err.println("Si e' verificato l'errore:
"+e.toString());
 }
}

public static void main(String args[]) { new
Lettore().leggi(); }

}
```

Compilate le classi ed eseguite nell'ordine la classe Scrittore e la classe Lettore. Per il nostro piccolo esempio è fondamentale che la classe Pugile (il nostro piccolo "contenitore") ed il file tito.jpg siano nella stessa directory.

CONCLUSIONI

Come premesso, il presente articolo rappresenta solo una piccola e leggera introduzione ad una tecnica di programmazione che può avere molti sviluppi interessanti. Si tratta di strumenti in grado di sollecitare la fantasia e la curiosità del programmatore, e questa è, a mio parere, una ulteriore buona ragione per guardarvi con attenzione.
Un suggerimento può essere quello di provare e riprovare seguendo le proprie intuizioni, allo scopo di fare pratica e di prepararsi al passo successivo, che sarà costituito dalla scrittura di una piccola applicazione client/server in grado di spedire un oggetto serializzato lungo una rete.

di Spirit
redazione@hackerjournal.it

EXPLOIT E DINTORNI

I PRINCIPALI WORM, TROJAN, MALWARE E ALTRE TIPOLOGIE DI ATTACCO, CHE SI SONO MESSE IN LUCE IN QUESTO ULTIMO MESE, NELLA NOSTRA RASSEGNA STAMPA DEDICATA AL MONDO DELLA SICUREZZA INFORMATICA.

Antivirus a rischio

La quantità di antivirus fittizi è andata calando negli ultimi tempi: i veri antivirus sono in grado di gestire i falsi creati dai cybercriminali e i tentativi di installarli sui computer stanno perdendo la loro efficacia. Ma i criminali informatici si sono inventati un altro modo per arrivare agli utenti: hanno infatti cominciato a lanciare i falsi antivirus non sui computer degli utenti bensì in Internet. In questo caso non si richiede più di scaricare il file sul computer, ma, senza dover aggirare la protezione antivirus, si raggiunge lo scopo di reindirizzare l'utente a un determinato sito in maniera più semplice. Nell'ultimo mese alcuni di questi nuovi "antivirus Internet" sono balzati in testa alle classifiche dei malware individuati nella rete. Uno di questi "antivirus", il Trojan.HTML. Fraud.ct. "inganna"generando una pagina Internet simile alla finestra "Risorse del computer" dei sistemi operativi Windows. Dopodiché tutto si svolge secondo il consueto copione: viene avviato un controllo fittizio del computer per rilevare la presenza di virus che vengono immediatamente "individuati". Quando l'utente acconsente all'eliminazione dei virus dal suo sistema, sul computer viene scaricato l'antivirus fittizio che invita l'utente ad acquistare la licenza (ovviamente è previsto che la "rimozione dei virus" dal computer abbia luogo solo previo pagamento). La maggior parte dei computer nei quali è stata rilevata la presenza di questo tipo di malware è localizzata geograficamente in paesi sviluppati, vale a dire negli Stati Uniti, in Canada, in Gran Bretagna, in Germania e in Francia. Nell'elenco rientra anche l'India, probabilmente perché lì risiedono molti utenti anglofoni.

Link camuffati

Di recente hanno acquisito una certa popolarità i servizi per accorciare gli URL. Tale popolarità è dovuta al fatto che in Twitter la lunghezza dei messaggi è limitata a 140 caratteri. L'utilizzo di tali servizi consente di occultare link infetti, opportunità che i cybercriminali non si lasciano certo scappare. In dicembre, nel corso di uno degli attacchi malware scagliati contro il servizio di microblog Twitter, nella pagina principale, nell'elenco dei temi che più interessano gli utenti, alcuni temi sono saliti artificiosamente di posizione, vale a dire con l'aiuto di malware. Tutti questi temi contenevano link sotto le mentite spoglie di servizi quali bit.ly, alturl.com ecc. Cliccando su questi link, l'utente veniva reindirizzato, per mezzo di alcuni redirector, alla pagina Web infetta dalla quale il computer scaricava senza dare nell'occhio il programma malware. Anche il servizio goo.gl di Google è stato utilizzato dai cybercriminali per diffondere i link maligni in Twitter all'inizio di dicembre. Un altro metodo utilizzato per mascherare i link dannosi è il mailing IM con messaggi di posta elettronica contenenti link di rimando alla pagina di Facebook nella quale si avverte l'utente che sta lasciando il sito del social network. Tuttavia il link era stato manipolato dai criminali in modo tale che quando l'utente, dopo aver cliccato sul link, premeva il pulsante "Continua" nella finestra per uscire da Facebook, veniva reindirizzato alla risorsa maligna.

TDSS, quando il gioco si fa duro...

Oltre all'organizzazione di attacchi fraudolenti in rete e ad attacchi non certo dei più complessi attuati tramite i social network, i cybercriminali lavorano anche all'"artiglieria pesante" dell'arsenale dei malware. Gli autori di uno dei malware più complessi al giorno d'oggi, il rootkit TDSS stanno continuando a perfezionarlo. In dicembre l'ultima versione del rootkit, la TDL-4, aveva cominciato a sfruttare la vulnerabilità CVE-2010-3338, scoperta nel luglio 2010 durante le ricerche compiute sul worm Stuxnet.

Trojan-Downloader.Java. OpenConnection

Di norma, questi malware vengono utilizzati dai cybercriminali nell'ultimo stadio dei download drive-by. Tuttavia per scaricare gli oggetti maligni sui computer degli utenti, non sfruttano delle vulnerabilità, ma il metodo OpenConnection della classe URL.
Per scaricare e lanciare il file maligno dal Web, tutti i rappresentanti della famiglia Trojan-Downloader.Java. OpenConnection non sfruttano delle vulnerabilità, bensì delle possibilità standard di Java. Per i malware scritti in linguaggio Java, questo metodo di download è attualmente uno dei principali. È facile supporre che la popolarità dei programmi maligni di questa famiglia continuerà ad aumentare fino a quando Oracle non chiuderà la possibilità di download dei file da essi sfruttata.



di Massimiliano Rinaldi
redazione@hackerjournal.it

KISMAC

E LA PASSWORD WI-FI E' SERVITA!



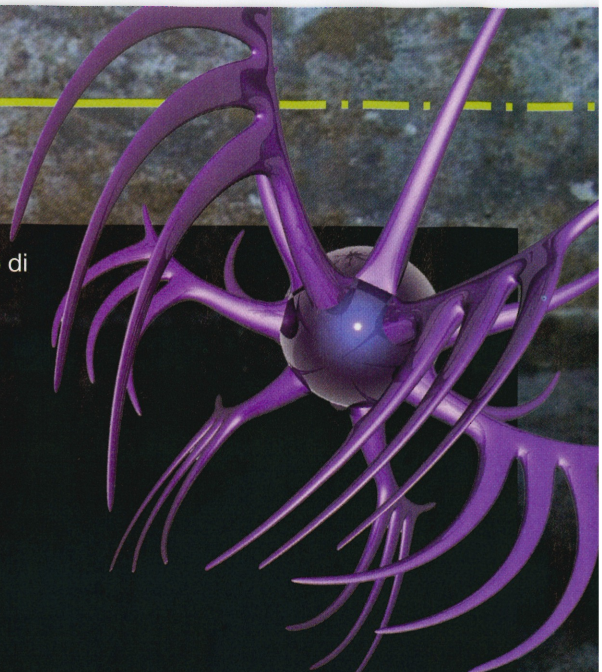
SCANNING

COME TROVARE LA PASSWORD DI ACCESSI INTERNET WI-FI PROTETTI IN POCHI SEMPLICI MOSSE...

Gli utenti Mac sono spesso un po' discriminati in tema di hacking. Da una parte possono a pieno titolo vantarsi di avere dei computer poco soggetti a Virus, Trojan, Malware e tutto quanto rientri nell'arsenale del perfetto hacker. Dall'altra sono, di fatto, sprovvisti loro stessi del suddetto arsenale: ovvero non hanno gli strumenti per cercare di realizzare qualche piccolo attacco, anche solo per testare l'efficacia della propria rete o delle difese collegate. Colpa della scarsa diffusione dei Mac rispetto ai PC/Windows, che rappresentano la stragrande maggioranza dei computer in commercio e, proprio per questo, finiscono per attirare l'attenzione dei malintenzionati. Ovvero, la troppa popolarità spesso può essere anche pericolosa, ma l'isolamento non favorisce il proliferare di idee e tecniche di sviluppo. Ma questo è un discorso fatto già altre volte e un po' ritrito. Semmai in questo articolo vogliamo segnalare un'eccezione che conferma la regola, ovvero un ottimo sniffer/scanner wi-fi per Mac che rappresenta una valida alternativa a Kismet e altri scanner disponibili in ambiente Windows/Linux: KisMAC. KisMAC è completamente gratuito, si può scaricare all'indirizzo http://trac.kismac-ng.org. Funziona benissimo coi portatili dotati di scheda airport integrata che consentono di spostarsi, in piena mobilità, alla ricerca di reti wi-fi attaccabili. Per testare l'efficacia di KisMAC abbiamo provato a utilizzarlo in ambiente esterno, con un MacBook Pro che si è trasformato in un potente scanner portatile.

QUALCHE SETTAGGIO

Prima di iniziare il nostro attacco dobbiamo selezionare i driver della scheda Wi-Fi integrata attraverso il pannello delle Preferenze (Figura 1). Nel nostro caso si tratta di una scheda Airport Extreme.



INIZIAMO

La versione utilizzata è la 0.3.2. Una volta lanciato KisMAC mostra la finestra principale che risulta assolutamente vuota (Figura 2). La nostra attività di scanning di pacchetti Wi-Fi, ovvero di intercettazione di pacchetti dati che transitano attraverso una rete wireless, non è ancora iniziata. Per avviarla dobbiamo premere Start Scan, nella parte inferiore destra della finestra principale. La manopola ancora più a destra inizierà una sorta di impercettibile movimento rotatorio. Segno che KisMAC sta lavorando per voi!
Ma prima di attivarsi KisMac chiede di autenticarsi come amministratore (Figura 3).

DATI DOPO DATI

KisMAC individua tutte le reti Wi-Fi nel raggio di azione della scheda Airport integrata e inizia a fare lo scanning dei pacchetti dati. Il nome delle reti disponibili appare nella colonna SSID. Qui per ovvi motivi li abbiamo cancellati. L'attività di raccolta dati deve procedere per diversi minuti. Infatti, per potere cercare di portare un qualsiasi tipo di attacco bisogna collezionare un quantitativo minimo di pacchetti dati.

UN ATTACCO DI PURA "FORZA BRUTA"

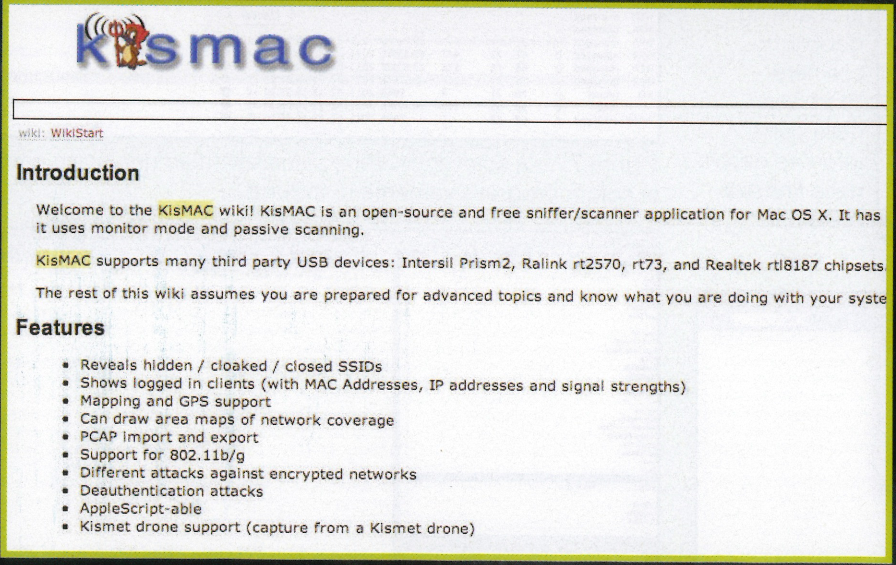
Come si può notare dalla nostra schermata (Figura 4) sono state individuate diverse reti, alcune con chiave di protezione WEP, altre con chiave di protezione WPA e WPA2. Evidentemente le reti WEP sono più vulnerabili ed è proprio su queste che concentreremo il nostro attacco. Dopo diversi minuti di intercettazione di pacchetti dati possiamo verificare se abbiamo elementi sufficienti per sferrare il nostro primo attacco. Proviamo a selezionare una rete Wi-Fi (tra quelle WEP) dal menu Network e proviamo a portare un attacco di pura forza bruta, ovvero un tipo di attacco che cerca di scovare la password forzando centinaia di combinazioni di lettere e parole al secondo.

```
Network>Bruteforce>alphanums
against 40-bit
```

In questo caso abbiamo scelto un attacco in grado di rivelare una password alfanumerica a 40 bit (Figura 5).
Se la quantità di dati è insufficiente per sferrare l'attacco comparirà un avviso che informa l'utente come sia necessario acquisire più dati prima di procedere. L'attacco può durare diversi minuti, la scansione viene visualizzata su una barra che si colora progressivamente di azzurro (come si può osservare dalle figure 6 e 7 presenti nella pagina successiva).

LA PASSWORD

Dopo circa un'ora KisMAC ha svelato la password di una delle reti WEP (Figura 8). Basta selezionare il pannello Proprietà (premendo il pulsante a forma di ingranaggio nella parte inferiore della finestra) per visualizzarla in corrispondenza della riga ASCII Key. Nel nostro caso era una banalissima password di sole cinque lettere, senza numeri, praticamente una delle password che gli esperti di sicurezza giudicano molto debole. In genere si consiglia di inserire non solo lettere ma anche numeri e di aumentare sensibilmente la lunghezza complessiva della password per aumentare il numero di combinazioni da forzare in un attacco "brute force". Naturalmente nella schermata abbiamo cancellato tutti i dati relativi alla rete Wi-Fi utilizzata per la nostra dimostrazione.
Nella pagina seguente trovate le schermate di tutto l'attacco con le relative descrizioni passo dopo passo.
Buono "sniffer" a tutti!



kismac

wiki: WikiStart

Introduction

Welcome to the KisMAC wiki! KisMAC is an open-source and free sniffer/scanner application for Mac OS X. It has it uses monitor mode and passive scanning.

KisMAC supports many third party USB devices: Intersil Prism2, Ralink rt2570, rt73, and Realtek rtl8187 chipsets.

The rest of this wiki assumes you are prepared for advanced topics and know what you are doing with your syste

Features

- Reveals hidden / cloaked / closed SSIDs
- Shows logged in clients (with MAC Addresses, IP addresses and signal strengths)
- Mapping and GPS support
- Can draw area maps of network coverage
- PCAP import and export
- Support for 802.11b/g
- Different attacks against encrypted networks
- Deauthentication attacks
- AppleScript-able
- Kismet drone support (capture from a Kismet drone)

KisMAC è completamente gratuito, si può scaricare all'indirizzo http://trac.kismac-ng.org.

Le fasi dell'attacco

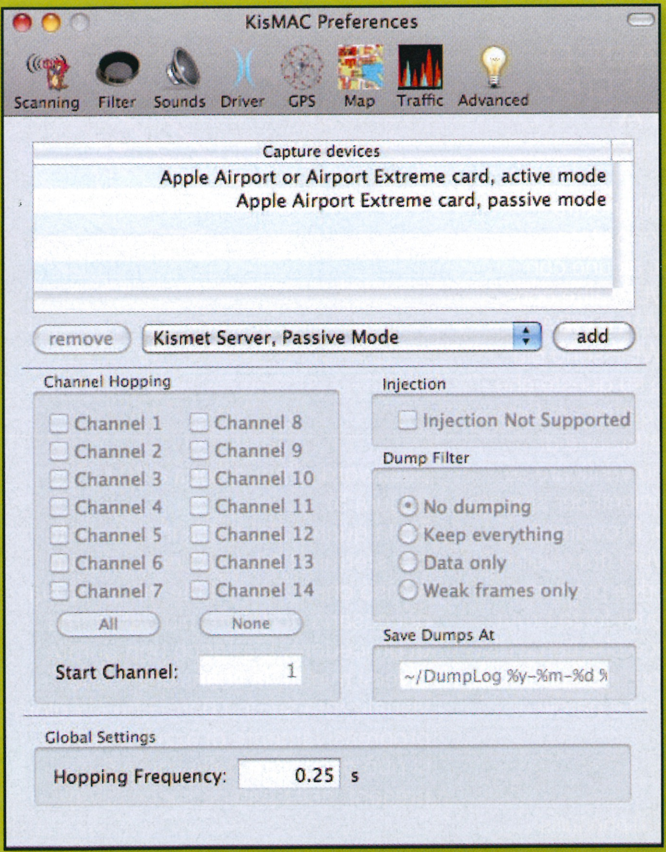


Figura 1: Prima di iniziare il nostro attacco dobbiamo selezionare i driver della scheda Wi-Fi integrata.

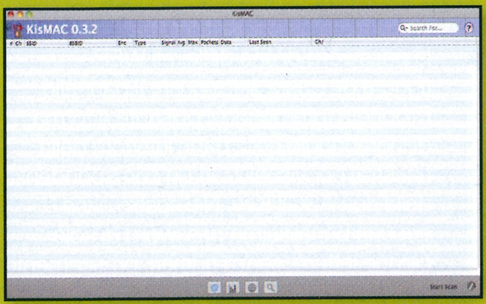


Figura 2: Per avviare lo scanning dobbiamo premere Start Scan, nella parte inferiore destra della finestra principale.

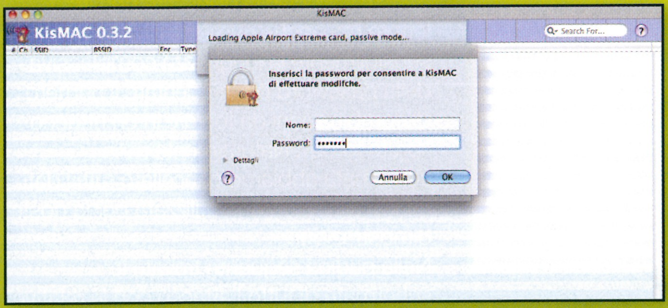


Figura 3: Prima di attivarsi KisMac chiede di autenticarsi come amministratore.

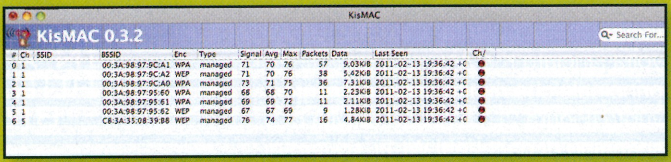


Figura 4: KisMAC individua tutte le reti Wi-Fi nella raggio di azione della scheda Airport e inizia a lo scanning.

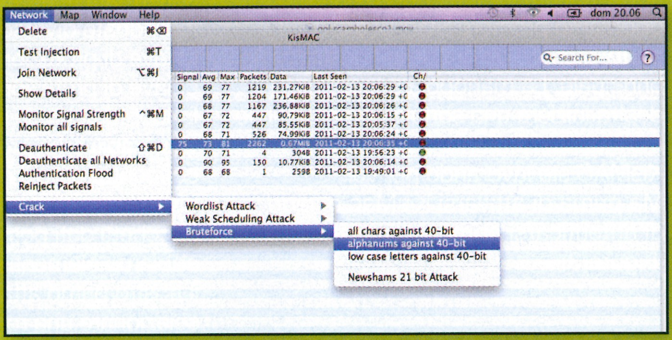


Figura 5: Selezioniamo dal menu: Network>Bruteforce> alphanums against 40-bit.

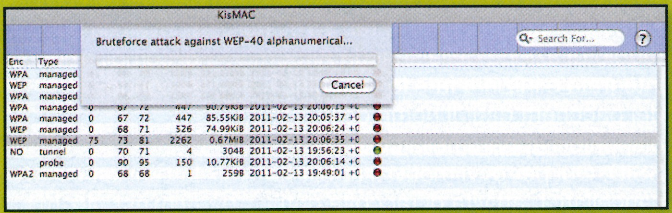


Figura 6: L'attacco può durare diversi minuti...

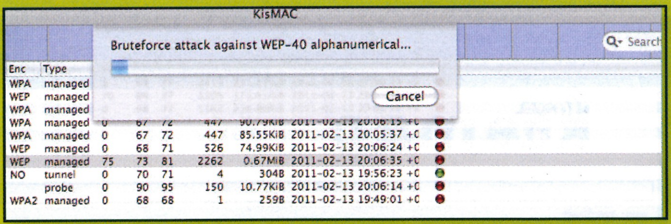


Figura 7: ...la scansione viene visualizzata su una barra che si colora progressivamente di azzurro.

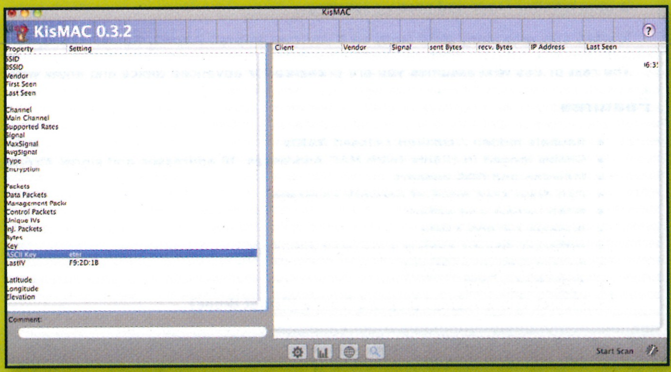
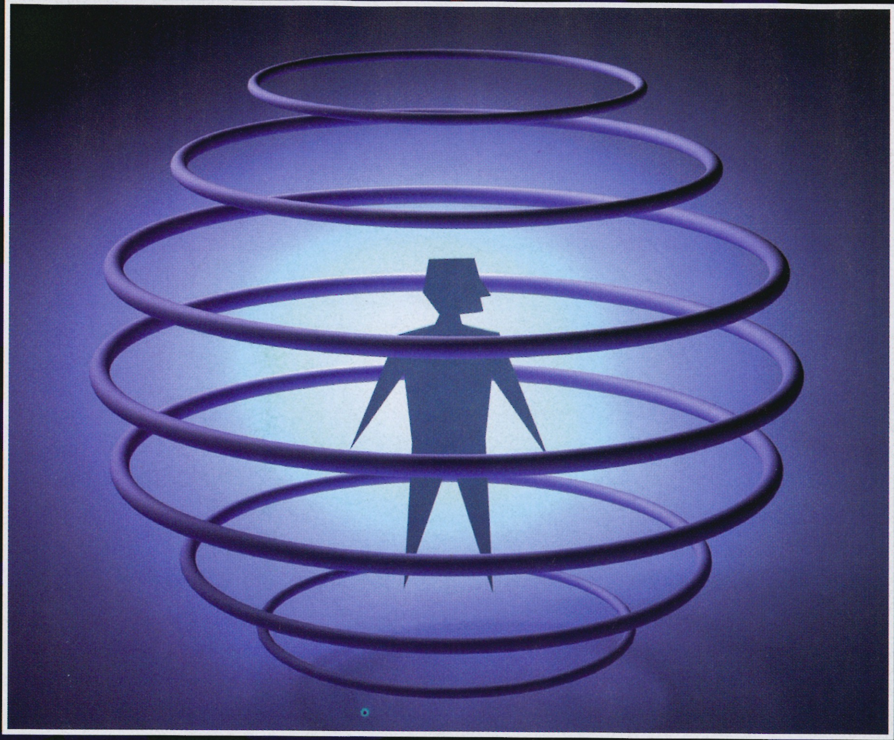


Figura 8: Dopo circa un'ora KisMAC ha svelato la password di una delle reti WEP.

di Alberto Lipari
redazione@hackerjournal.it

LE VIOLAZIONI NEGLI ACCESSI DI SISTEMA



SICUREZZA
PER SCOPRIRE SE UN COMPUTER SIA STATO OGGETTO DI TENTATIVI DI ACCESSO INDESIDERATO ESISTONO DIVERSI STRUMENTI, COME LOGCHECK...

Uno dei modi per verificare, ed eventualmente scoprire, se un computer sia stato oggetto di tentativi indesiderati d'accesso, è quello di ricercare le informazioni nei file di log, accertandosi che nessuno si sia intromesso nel nostro sistema, nell'assenza di manomissioni agli stessi file.
Cercare eventi di questo tipo nei vari file è un operazione laboriosa, che può richiedere ore, se non qualche giorno, a seconda delle dimensioni dei log e della tipologia di eventi che ricerchiamo. Per tale scopo sarebbe utile poter disporre di un programma che, eseguito ciclicamente, estragga dalle informazioni contenute nei vari file di log del sistema solo quelle relative ad eventi anomali o collegati a tentativi d'accesso non autorizzati e li invii ad un indirizzo di posta (generalmente all'amministratore del sistema), dopo averli ordinati e organizzati per tipologia.

LOGCHECK: LO SCOVA INTRUSI

Per le finalità appena descritte utilizzeremo uno script di shell che si chiama logcheck, sviluppato da Craig Rowland. Le moderne distribuzioni contengono questo strumento, ma per l'ultima versione si consiglia di visitare siti come RPM Find – http://rpmfind.net. Oltre alle distribuzioni Gnu/Linux, è possibile installare ed utilizzare questo software per altri sistemi operativi del ceppo Unix (FreeBSD, BSDI, Sun, HPUX, Digital-OSF/1, Irix) scaricando quanto serve da:

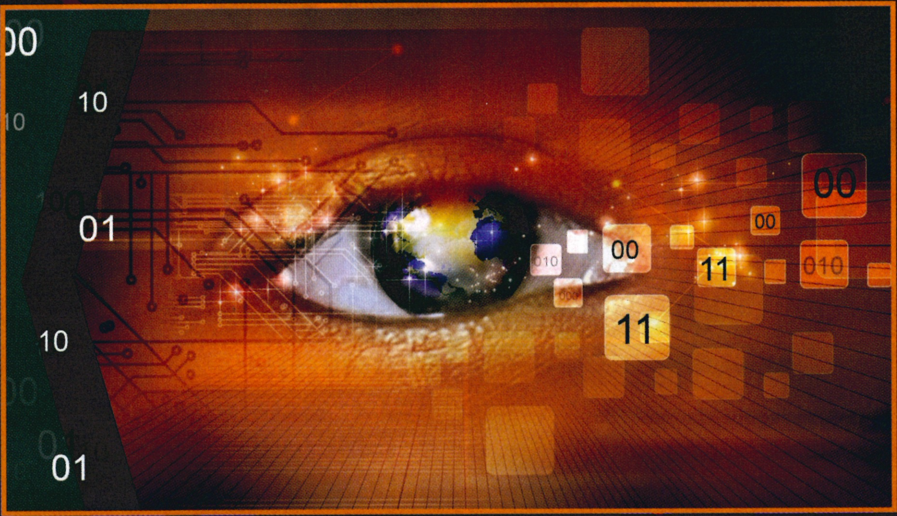
`http://logcheck.org/`

Per l'installazione è necessario essere utente root e digitare:

$ rpm –ivh logcheck-1.3.13-4.i386.rpm

Se vogliamo visualizzare la lista dei file che sono stati installati sulla nostra macchina, occorre eseguire il comando "rpm –qil logcheck". I principali file che saranno listati devono essere i seguenti:

```
/usr/share/doc/logcheck-1.3.13
/var/logcheck
```




```
/usr/bin/logtail
/etc/logcheck/hacking
/etc/logcheck/violations
/etc/logcheck/violations.ignore
/etc/logcheck/ignore
/usr/bin/logcheck.sh
/etc/cron.hourly/logcheck
```

Esaminiamo quelli più importanti.

`/usr/share/doc/logcheck-1.3.13`

contiene i file di documentazione relativi al pacchetto (README, INSTALL, CHANGES, ecc.);

`/var/logcheck`

è utilizzata come directory d'appoggio del file di log temporaneo, creato durante l'esecuzione di logcheck; questo file è costituito dall'accodamento di tutti i log di sistema che vogliamo controllare ed è cancellato al termine dell'esecuzione dello script;

`/usr/bin/logtail`

logtail è un programma, eseguito all'interno della script logcheck.sh, in grado di memorizzare il puntatore relativo all'ultima riga del file di log esaminato, ed è possibile utilizzarlo anche per altri file di tipo testuale, nel caso vi serva una funzione di questo tipo. Durante le successive esecuzioni dello script, le righe dei file di log che sono già state lette in precedenza, saranno scartate. Per ogni file di log sarà creato, nella stessa directory, un corrispondente file "########.offset", dove ####### è lo stesso nome del file di log esaminato; questo file conterrà un valore numerico riferito all'ultima riga letta;

`/etc/logcheck/hacking`

contiene stringhe di caratteri che, se trovate nei file di log, sono riportate nel messaggio di posta elettronica inviato e sono presentate alla fine in una sezione con il titolo "Active System Attack Alerts";

`/etc/logcheck/violations`

in questo file troviamo parole e stringhe che in genere sono contenute in eventi considerati "negativi"; record di log contenenti parole come "denied" e "refused" attivano durante l'esecuzione di logcheck righe di report nella sezione "Security Violations".

E' possibile in ogni caso che tali eventi, anche se segnalati, possano essere accettati (ad esempio un utente che ha sbagliato la propria password durante la procedura d'accesso al sistema);

`/etc/logcheck/violations.ignore`

supponiamo che una stessa stringa di caratteri sia contenuta in due diversi record di log, ma che solo uno di questi debba essere riportato come violazione al sistema. Possiamo utilizzare il file violations.ignore allo scopo, chiarendo questo concetto l'esempio riportato di seguito.

VIOLATIONS. IGNORE IN PRATICA

Poniamo di avere due record di log:

```
Sep 10 21:00:08 localhost
sendmail[23123]: GAA03745:
to=marvin, ctladdr=root
(0/0), delay=00:00:03,
xdelay=00:00:02, mailer=local,
stat=refused

Sep 17 17:17:17 localhost rshd:
refused connect from guestuser@
localhost:1490
```

Poiché vogliamo che solo la seconda riga di log (quella delle ore 17:17:17) debba attivare un allarme, allora inseriremo la stringa "refused" nel file violations e "mailer=local, stat=refused" nel file violations.ignore

`/etc/logcheck/ignore`

contiene le stringhe da ignorare e da non indicare in nessuna sezione del report finale; tutte le righe di log contenenti parole che non saranno trovate in nessuno dei quattro file appena visti, utilizzati da logcheck per filtrare i log (hacking, violations, violations.ignore, ignore), saranno aggiunte al report finale all'interno della terza sezione identificata con "Unusual System Events";

`/usr/bin/logcheck.sh`

questo script shell è il fulcro delle attività di controllo e contiene al suo interno varie sezioni contenenti le definizioni di variabili relative ai diversi sistemi operativi dove logcheck può essere eseguito. Se vogliamo quindi eseguire logcheck.sh su una macchina con un sistema operativo diverso da Linux, occorre rendere attive (non commentate) solo le righe relative a quel particolare sistema perché, rispetto a quest'ultimo, i comandi utilizzati all'interno dello script possono variare. Ad esempio, in logcheck.sh è definita la variabile MAIL, che può assumere i valori "mail", "mailx", "Mail", rispettivamente per i sistemi operativi Linux, HPUX, Digital-OSF/1.

SCRIPT MANUALE O AUTOMATICO

Lo script logcheck.sh può essere eseguito manualmente, oppure in modalità automatica, inserendo nella crontab (tabella dei comandi/programmi eseguiti a tempo dal programma cron) quanto segue:

```
10 07 * * * /usr/bin/logcheck.sh
```

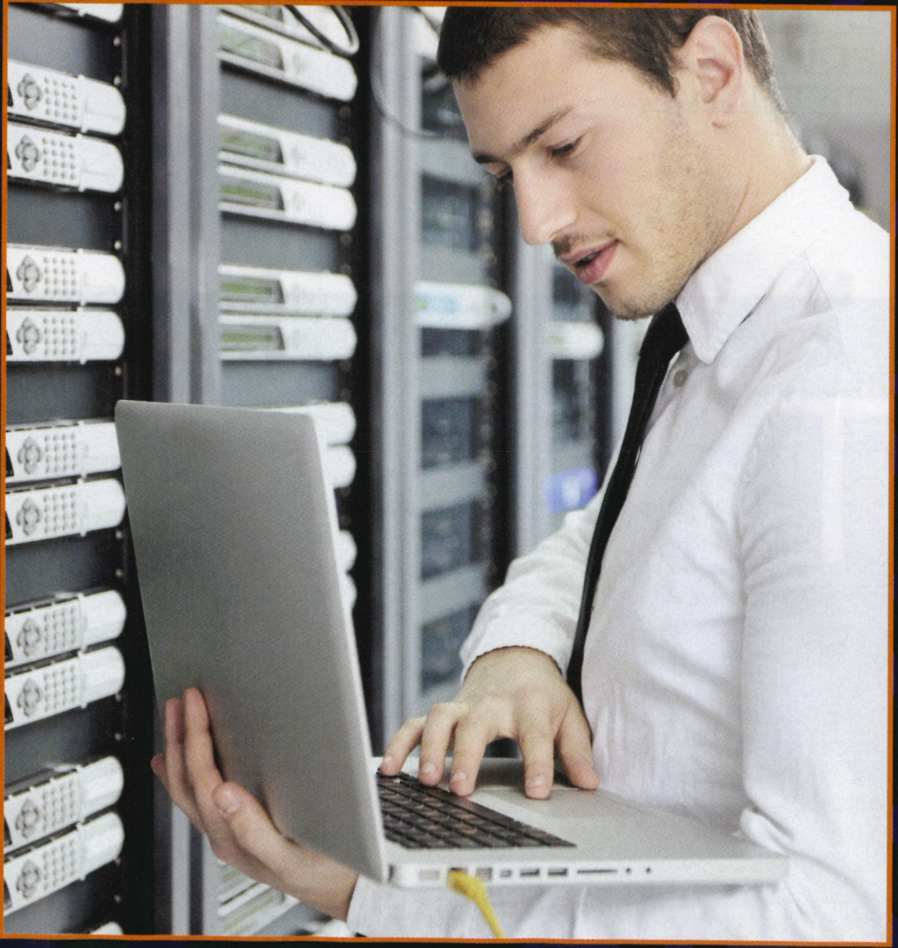
(per una dettagliata spiegazione dei possibili valori che possono essere assegnati ai campi di crontab usate il comando "man 5 crontab").
La sequenza logica delle operazioni effettuate dallo script logcheck.sh si può riassumere coi seguenti punti:
- è eseguito il programma logtail che accoda i file di log in un unico file temporaneo, prelevando solo le informazioni non ancora trattate da precedenti esecuzioni di logcheck;
- per ogni riga del file di log temporaneo creato, è eseguita una scansione per la ricerca delle stringhe uguali a quelle contenute nel file "hacking", generando, se trovate, la sezione di report intitolata "Active System Attack Alerts";
- per ogni riga del file di log è eseguita una scansione per la ricerca delle stringhe uguali a quelle contenute nel file "violations", scartando, per quelle con confronto positivo, le righe che contengono stringhe di caratteri uguali a quelle contenute in "violations.ignore" e generando la sezione di report intitolata "Security Violations". Circa la relazione tra "violations" e "violations.ignore", rivedete l'"esempio fatto quando abbiamo parlato di questi due file. La terza ed ultima sezione del report, "Unusual System Events", contiene tutte le restanti informazioni di log che non hanno trovato corrispondenza di stringhe nel file ignore, precedentemente visto. Il report sarà spedito all'indirizzo di posta definito nella variabile SYSADMIN.

```
Active System Attack Alerts
=-=-=-=-=-=-=-=-=-=-=-=-=-=
Sep  6 20:43:47 localhost sendmail[23123]: g84Ihl223123: localhost [192.168.0
.99]: vrfy root [rejected]

Security Violations
=-=-=-=-=-=-=-=-=-=
Sep  6 21:49:12 localhost su(pam_unix)[1233]: authentication failure; logname
=marvin uid=500 euid=0 tty= ruser=marvin rhost=  user=root

Unusual System Events
=-=-=-=-=-=-=-=-=-=-=
Sep  6 21:47:06 localhost kde(pam_unix)[926]: session opened for user marvin
by (uid=0)
Sep  6 21:49:22 localhost su(pam_unix)[1234]: session opened for user root by
 marvin(uid=500)
Sep  6 21:50:56 localhost su(pam_unix)[1294]: session opened for user marvin
by marvin(uid=500)
```

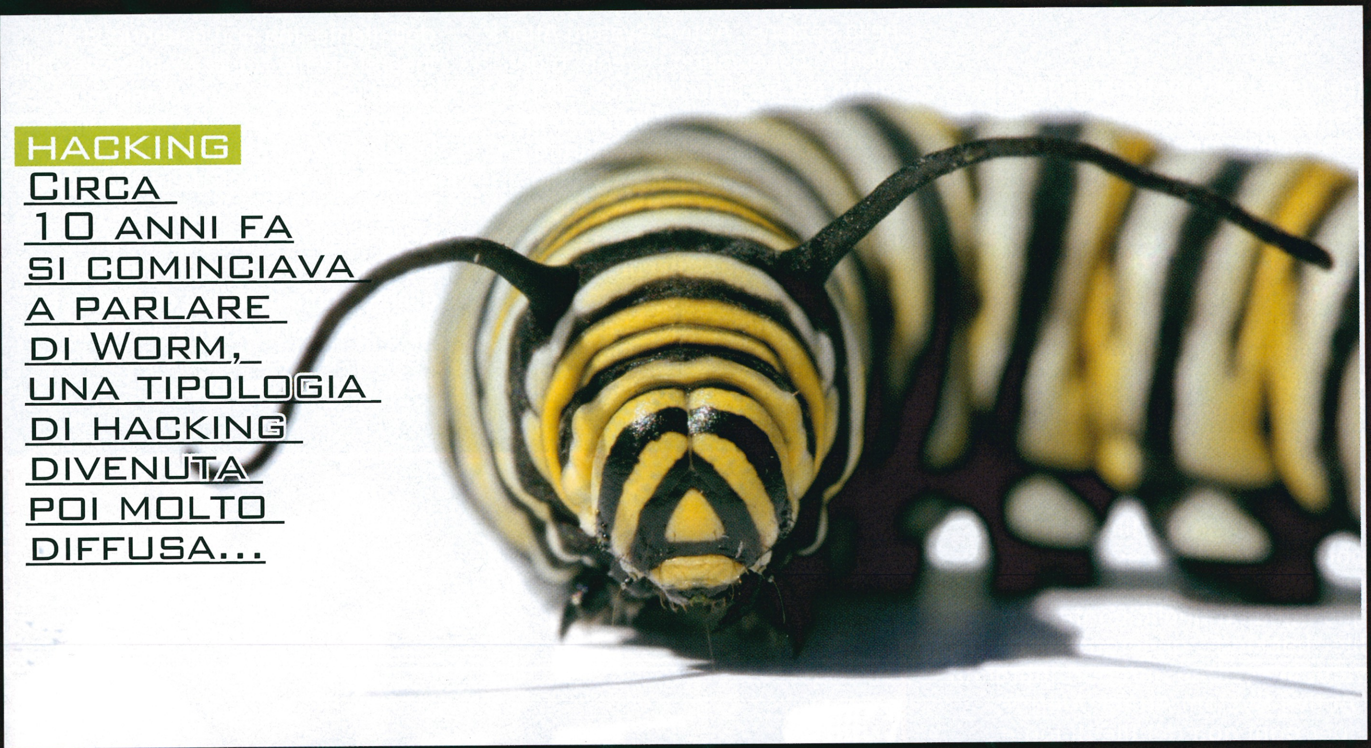
Osservando l'immagine precedente, nella sezione "Active System Attack Alerts", osservando l'evento registrato nei file di log alle ore 20:43:47 è evidente come dall"indirizzo Ip 192.168.0.99 sia stato impartito un comando sendmail di "vrfy"; ciò potrebbe indicare l'attivazione di un processo telnet sulla porta 25 (smtp) per scopi illeciti. L'evento delle ore 21:49:12, nella sezione "Security Violations", denuncia il fallito tentativo da parte dell'utente marvin di impartire il comando su (superuser) per ottenere i privilegi dell'amministratore di sistema; in questo caso l'utente marvin aveva sbagliato la password di root. Nella sezione "Unusual System Events", infine, si rilevano tre record di log relativi ad operazioni d'autenticazione; sono eventi di normale routine durante l'attività quotidiana dell'utente, ma non essendo stati inseriti nel file "ignore" utilizzato dalla script logcheck, sono comunque inseriti nel report. Concludendo, logcheck sicuramente può essere uno degli strumenti da utilizzare per implementare una politica di controllo relativa alla sicurezza di un sistema, ma i suoi report devono essere continuamente esaminati e se sospettiamo attività illecita, la frequenza d'esecuzione dello script deve essere aumentata.





di Massimiliano Rinaldi
redazione@hackerjournal.it

DAL WORM AI GIORNI NOSTRI



HACKING

CIRCA 10 ANNI FA SI COMINCIAVA A PARLARE DI WORM, UNA TIPOLOGIA DI HACKING DIVENUTA POI MOLTO DIFFUSA...

Tutto è partito dal virus, la prima vera minaccia informatica di cui si è avuta notizia molto, ma molto tempo fa. Era un'epoca quella, in cui Internet si affacciava nella vita quotidiana, timidamente, pur facendo già intravedere grandi potenzialità. A dire il vero già la minaccia di avere un virus nel proprio computer metteva una certa agitazione, figuriamoci poi quando si è scoperto, di lì a poco, che la minaccia non era circoscritta a questa sorta di sintomatologia epidemiologica informatica. C'era di peggio: il verme, o worm...
La differenza sostanziale tra virus e worm è che il primo infetta il sistema copiandosi in programmi sani per poi riprodursi e creare danni, come la cancellazione di informazioni. Inoltre, rispetto al worm, il virus è autosufficiente e non ha bisogno di nessun programma sano per diffondersi. Altra differenza significativa è che il worm viaggia in Rete utilizzando i più diffusi protocolli, azione impossibile per un normale virus.
In questo articolo ripercorremo le "gesta" dei più importanti worm che hanno segnato gli albori della storia di Internet, illustrandone la modalità di penetrazione, da Morris a Ramen a Lion, passando per il pericoloso Adore fino al benefico worm Cheese e al"simpatico" DoS.Storm.Worm.

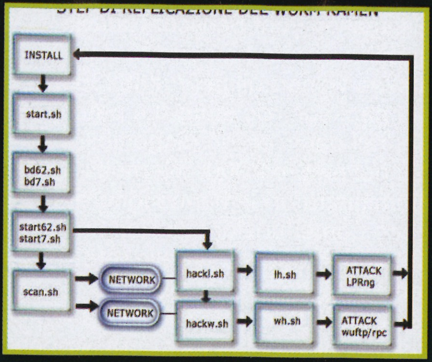
IL PRIMO WORM: MORRIS 1988

Tutto ebbe inizio nel neanche tanto lontano 1988, quando, per la prima volta, il mondo informatico prestò attenzione ai problemi della sicurezza su Internet. Fu grazie ad uno studente universitario di Cornell, Robert Tappan Morris, il quale scrisse un worm che, secondo la sua testimonianza, gli sfuggì di mano viaggiando autonomamente in Rete e causando il crash di migliaia di sistemi. Com'è stato possibile un simile evento? Morris si accorse che i server Unix, nella versione di Berkeley, avevano

due problemi di sicurezza il cui effetto era l'accesso al sistema con i permessi di superutente. Alla luce di questa scoperta, ebbe la brillante idea di scrivere un programma che avrebbe sfruttato queste due falle autoriproducendosi e infettando nuovi sistemi. La "creatura" di Morris si componeva di due programmi: il bootstrap e il worm vero e proprio. Il primo comprendeva una novantina di linee di codice già compilato che veniva eseguito nel sistema sottoposto all'attacco; una volta lanciato, si collegava alla macchina da cui proveniva e caricava il worm principale eseguendolo. Il passo successivo era quello di controllare le tabelle di instradamento del sistema alla ricerca di nuove macchine da infettare.
Per farlo, il worm di Morris tentava tre vie principali. Inizialmente, cercava di eseguire una shell remota tramite il noto comando rsh: a quei tempi, i sistemi si "fidavano" delle altre macchine non richiedendo alcuna autenticazione; se andava a buon fine, la shell caricava il worm per poi eseguirlo. Oggi, una situazione del genere sarebbe impensabile.
Il secondo metodo sfruttava un programma che girava sui sistemi Bsd chiamato finger, il quale permette la visualizzazione delle informazioni riguardanti un utente di sistema da remoto. Oggi tale demone è presente ma poco usato. Tra le informazioni usate a quei tempi, si poteva conoscere il nome della persona, il tipo di collegamento, l'indirizzo di casa e il numero di telefono: insomma, l'equivalente dell'attuale rubrica, e negli ambienti universitari e di ricerca queste informazioni erano di vitale importanza. Tale demone era sempre attivo e pronto a rispondere alle richieste degli utenti Internet, ma aveva un problema: infatti se la richiesta conteneva una stringa speciale di 536 byte, il demone andava in buffer overflow eseguendo le informazioni contenute in essa, normalmente si tentava la richiesta di una shell /bin/sh che il sistema offriva sistematicamente. L'ultimo metodo di attacco riguardava un problema del sistema di posta Sendmail: la configurazione della posta ai quei tempi era talmente complicata che gli sviluppatori di Sendmail avevano incorporato un sistema di debug per verificarne il corretto funzionamento.
Il problema derivava dal fatto che qualsiasi utente poteva inserire un insieme di comandi al posto del campo del destinatario (recipient) che Sendmail eseguiva con i permessi di superutente. Dunque, sono immaginabili le conseguenze di tale azione, che, peraltro, non richiedeva nessun tipo di autenticazione.
Dopo essere riuscito a penetrare nel sistema, il worm cercava di scoprire le password dell'utente: ogni password individuata permetteva al worm di connettersi a tutte le macchine su cui il proprietario aveva un account e, di conseguenza, replicarsi senza utilizzare le modalità descritte in precedenza. Alla fine, il risultato dei danni prodotti dal worm, fu l'arresto di Morris: egli spiegò agli agenti federali di non aver avuto intenzione di creare danni, si trattava semplicemente un test sfuggito al controllo. Ma non venne molto "capito". Del resto la ricerca in ogni epoca ha i suoi martiri.

RAMEN

La diffusione del worm avveniva sfruttando buchi di sicurezza dei sistemi. Un pugno di amministratori si era accorto che, fin dal rilascio, le versioni di Linux RedHat 6.2 e 7.0 soffrivano di alcuni bachi nei package di default di sistema. Un problema che ha così permesso la massiccia diffusione di un nuovo worm chiamato Ramen, una collezione di script studiati appositamente per sfruttare le tre principali vulnerabilità che compromettono l'account root. La prima, wu-ftpd (port 21/tcp), era un pacchetto che si occupava di attivare il demone necessario per far funzionare il server ftp Washington University. La vulnerabilità (Vu#29823) consisteva nella funzione site exec che, tramite una stringa non convenzionale, permetteva l'esecuzione di codice come utente root. Per quanto riguarda la seconda vulnerabilità, rpc.statd (port 111/udp), si trattava di un demone che faceva parte del pacchetto nfs-utils il cui compito è quello di eseguire il controllo del file locking sui sistemi in cui nfs è attivo. La vulnerabilità (Vu#34043) comprometteva l'account di root che, attraverso una stringa e una chiamata alla funzione syslog(), permetteva a Ramen di eseguire codice sul sistema remoto.
La terza, LPRng (port 515/tcp), è stata chiamata dagli sviluppatori Next



Lo schema di auto-replicazione del worm Ramen.



di Massimiliano Rinaldi
redazione@hackerjournal.it

Generation Print Service. Tuttavia, appena uscita, ha sofferto delle stesse vulnerabilità (Vu#382365) del package rpc.statd. Le patch per tali bachi erano disponibili dal novembre 2000, ma molti amministratori le hanno ignorate…
Ecco come funzionava Ramen: attraverso uno scan della rete, tentava l'accesso ai sistemi remoti sfruttando in sequenza le vulnerabilità descritte sopra. Appena riusciva a entrare nel sistema, le azioni compiute erano molteplici. Iniziava con la creazione di una directory in /usr/src/.poop dove replicava se stesso e, a questo punto, partiva lo script start.sh che si occupava di eseguire le successive azioni. Dunque, passava alla sostituzione di tutti i file index.html con uno nuovo su cui appariva la scritta: "RameN Crew Hackers looooooooooooove noodles". Eseguite queste operazioni, passava alla creazione del file /usr/src/.poop/myip che conteneva l'indirizzo Ip del sistema e aggiungeva il codice in /etc/rc.d/rc.sysinit che si occupava di eseguire uno scan della rete al successivo riavvio in modo da individuare nuovi sistemi da attaccare. Quindi, inseriva in /etc/xinetd.conf o /etc/inetd.conf una stringa che attivava un nuovo demone sulla porta 27374 chiamato asp, usato da Ramen per copiarsi su un nuovo sistema. In seguito, provvedeva all'eliminazione del file /etc/hosts.deny, blocca i servizio ftp, rpc.statd (sui sistemi RedHat 6.2) e lpd (sui sistemi RedHat 7.0) con successiva sostituzione di un file vuoto in modo da evitare nuovi attacchi. Infine, inviava due mail agli indirizzi gb31337@hotmail.com e gb31337@yahoo.com per avvisare della compromissione di un nuovo sistema. Dopo queste azioni, Ramen eseguiva al riavvio uno scan della rete, alla ricerca di nuovi sistemi da infettare copiando se stesso sul nuovo sistema attraverso la porta 27374 del sistema precedente e così via. I sistemi compromessi da Ramen sono stati migliaia Il worm non permettava ai servizi responsabili dei problemi di riavviarsi: questo per evitare di attaccare sistemi già infettati.

Nome File	Descrizione
asp	Script che modifica la configurazione di avvio dei servizi del sistema.
asp62 - asp7	Demoni per una root shell nella porta 27374
bd62.sh - bd7.sh	Script per l'avvio del demone asp62 o asp7
hackl.sh - hackw.sh	Exploit per la ricerca di sistemi 7.0 e 6.2 analizzati dal syscan output file
l62 - l7	Exploit per LPRng
index.html	Pagina html da sostituire a quelle presenti nel sistema
start.sh	Script per azioni varie come la sostituzione dell'index, o lancio di attacchi
scan.sh	Scansione della rete alla ricerca di host vulnerabili

Tabella 1 - Principali script utilizzati da Ramen per infettare il sistema e replicarsi.

LA MINACCIA: LION E ADORE

Ramen, come era prevedibile, ha aperto la strada a nuovi worm che sfruttavano più o meno le stesse vulnerabilità. Ma mentre Ramen non creava danno ai sistemi, Lion e Adore non hanno utilizzato la stessa "gentilezza". L'istituto di ricerca The System Administration, Networking and Security (Sans) ha giudicato Lion l'evoluzione peggiore di Ramen perché attaccava sistemi in cui era presente la versione bacata di Bind, il server dns più usato al mondo.
Il worm Lion si diffondeva nel sistema attraverso un programma di nome randb: lo script eseguiva uno scan della rete alla ricerca di server Bind vulnerabili e, una volta trovato un potenziale bersaglio, vi si trasferiva utilizzando un famoso exploit name. Fatto questo, Lion installava un rootkit di nome t0rn e sostituiva diversi binari come find, ls e in.telnetd. Inoltre spediva i file /etc/passwd e /etc/shadow ad alcuni indirizzi di posta.
Il worm quindi creava delle backdoor di tipo root shell nelle porte 60008/tcp e 33567/tcp e installava un trojan nel server Ssh che stava in ascolto sulla porta 33568/tcp. Come se non bastasse, si preoccupava di cancellare completamente il demone Syslogd, così che nessun accesso al sistema potesse più essere registrato.

Adore, conosciuto anche come RedWorm, è stato anch'esso un worm "molto popolare". Non meno pericoloso di Lion, secondo gli esperti di sicurezza girava in rete dall'aprile 2001 e aveva la caratteristica di sfruttare tutte le quattro vulnerabilità descritte in precedenza .
Adore, quando riusciva ad installarsi su un sistema, era facilmente individuabile perché effettuava scansioni su tutta la classe B della rete che puntano alla porta 515 degli host. Quando il worm riusciva a trovare una macchina vulnerabile, si connetteva ad un server localizzato in Cina e trasferiva la parte principale del worm sull'host compromesso, all'interno della directory /usr/local/bin/lib. Qui eseguiva lo script start.sh. Fatto questo, la sua funzione di stealthing, implementata mediante la sostituzione del comando /bin/ps con uno modificato, gli permetteva di agire indisturbato mentre rimpiazzava il demone /sbin/klogd con l'unico scopo di aprire una shell che accettava connessioni sulla porta 65535.
Come Lion, aveva la capacità di spedire i file delle password via mail a quattro indirizzi di posta elettronica e copriva tutti i suoi processi usando uno script attivato via crontab e residente in /etc/cron.daily/0anacron.

IL BENEVOLO CHEESE WORM

Sempre nello stesso periodo di Lion, qualcuno ha avuto l'idea di creare un worm che combattesse Lion: avete

capito bene, sembrava di giocare a guardie e ladri (come del resto avviene anche oggi e non solo nel settore informatico), con l'irruzione di un nuovo supereroe, come si conviene in questi casi. Il difensore del pinguino (Linux), al secolo Cheese Worm. Il fatto che fosse "un verme" forse sminuisce un po' la portata del suo operato e rende difficile comunicare una bella immagine positiva, ma Cheese Worm è stato un vero e proprio paladino della rete, alla faccia di Spider Man.
Il suo scopo era quello cercare in rete tutti i sistemi infetti da Lion e chiudere le backdoor che questo creava. Per fare questo, Cheese eseguiva la scansione di una serie di indirizzi Ip alla ricerca delle backdoor (come la 10008) lasciate da Lion. Tramite queste porte, Cheese entrava nel sistema e si installava in /tmp/.cheese chiudendo le backdoor e ripartendo per una nuova destinazione.

Un altro worm famoso è stato: Dos. Storm.Worm: l'anti Microsoft. Questo worm era un dei più strani perché il suo obiettivo era sì quello di replicarsi, ma per lanciare un attacco ai server della Microsoft. La sua diffusione risale al giugno 2001 ed era stato appositamente studiato per aggredire i server web di WindowsNT/2K sfruttando un vecchio baco dei sistemi Iis (Internet Information Server). Tale vulnerabilità era nota da tempo (ottobre 2000), ma, ignorata, come al solito, dagli amministratori di sistema. Ma come agiva Dos.Storm? Una volta penetrato nel sistema attraverso la vulnerabilità conosciuta come Web server folder directory traversal, inviava migliaia di mail all'indirizzo gates@microsoft.com tentando di intasare la posta con il testo "Fuck you". Non contento, cominciava un pesante attacco Dos (Denial of Service) ai server gestiti dalla Microsoft.

CONCLUDENDO...

Come possiamo vedere, la vita dell'amministratore di sistema non è semplice, sommerso com'è da problematiche di configurazione dei servizi che spesso rubano il tempo alle problematiche, altrettanto delicate, legate alla sicurezza. I worm sono un argomento molto affascinante, e si sono molto evoluti nel corso del tempo costituendo, oggi, uno dei veicoli più utilizzati per commettere attività illecite in Internet. Se agli inizi il worm era soprattutto creato per scopi dimostrativi e sperimentali, oggi rappresenta un'arma in mano a vere e proprie organizzazioni dedite al cyber crimine,
E non è finita: si sta iniziano a parlare di worm dotati di un'intelligenza artificiale. Dunque, occhio e mano alle regole di Ipchains.



di Luca Buti
redazione@hackerjournal.it

ARRICCHIRSI COL CYBER CRIMINE

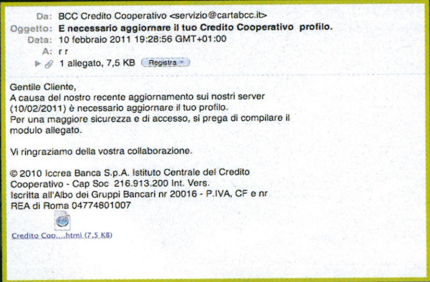


CYBER CRIMINE

QUANTO GUADAGNA UNO SPAMMER? E UN PHISHER? VEDIAMO QUANTO RENDONO LE PROFESSIONI ILLECITE IN INTERNET.

Tecnicamente le figure dello Spammer e del Phisher possono coesistere o rappresentare due figure separate. Lo Spammer si preoccupa di inviare mail dai contenuti più disperati sfruttando il suo enorme data base di indirizzi procurati in svario modo. Quindi può essere pagato da un'organizzazione per inviare, ad esempio, messaggi con link fraudolenti, come quelli che invitano a verificare l'account del proprio conto on-line "bloccato per motivi di sicurezza", oppure può guadagnare egli stesso dai proventi della truffa, e quindi essere anche phisher, che derivano dagli utenti, pochi, che cliccano sui link.

Da una ricerca svolta dall'Università della California, Berkley, UC e da quella di San Diego è stato dimostrato che servono almeno 12 milioni e mezzo di e-mail "spazzatura" per avere un minimo di ritorno in termini di click sui link fraudolenti e avviare in modo conveniente un'attività di phishing. Per ottenere risultati attendibili i ricercatori hanno messo in piedi una rete nella quale dei computer emulavano l'invio di e-mail spam con 350 milioni di messaggi spazzatura in soli 26 giorni. Da questo test di simulazione si è arrivati alla conclusione che solamente lo 0.00001% clicca sui link fraudolenti. La prima riflessione è che tale risultato potrebbe davvero sembrare irrisorio per gli spammer/phisher che, invece, considerata la mole di spam inviato, possono arrivare a guadagnare almeno 8.000 dollari al giorno. Davvero una bella cifra!



Da: BCC Credito Cooperativo <servizio@cartabcc.it>
Oggetto: **E necessario aggiornare il tuo Credito Cooperativo profilo.**
Data: 10 febbraio 2011 19:28:56 GMT+01:00
A: r r
1 allegato, 7,5 KB Registra

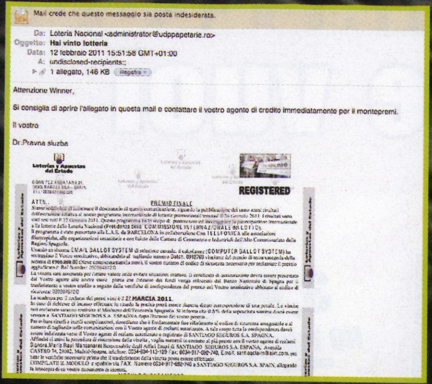
Gentile Cliente,
A causa del nostro recente aggiornamento sui nostri server (10/02/2011) è necessario aggiornare il tuo profilo.
Per una maggiore sicurezza e di accesso, si prega di compilare il modulo allegato.

Vi ringraziamo della vostra collaborazione.

© 2010 Iccrea Banca S.p.A. Istituto Centrale del Credito Cooperativo - Cap Soc 216.913.200 Int. Vers.
Iscritta all'Albo dei Gruppi Bancari nr 20016 - P.IVA, CF e nr REA di Roma 04774801007

Credito Coo....html (7,5 KB)

Gli avvisi di conto corrente bloccato o da aggiornare rappresentano uno degli approcci preferiti dei phisher.


Mail crede che questo messaggio sia posta indesiderata.

Da: Loteria Nacional <administrator@udppapeterie.ro>
Oggetto: Hai vinto lotteria
Data: 12 febbraio 2011 15:51:58 GMT+01:00
A: undisclosed-recipients:;
1 allegato, 146 KB

Attenzione Winner,

Si consiglia di aprire l'allegato in questa mail e contattare il vostro agente di credito immediatamente per il montepremi.

Il vostro

Dr.Pravna sluzba

Loterías y Apuestas del Estado

REGISTERED

ATTN.,

PREMIO FINALE

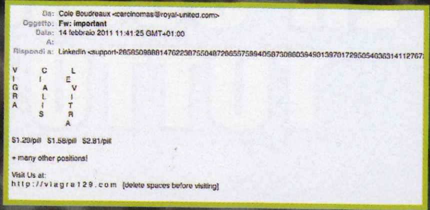
...anche le vincite alla lotteria vanno forte!

LA DIFFICILE ARTE DELLO SPAM

Diventare uno Spammer di successo, comunque, è tutt'altro che facile. La competizione è serrata, ci sono migliaia di spammer agguerriti sulla piazza che cercano di dividersi il ricco bottino derivante dai proventi.
Per emergere ci vuole una certa "creatività", i messaggi per essere cliccati (è sulla percentuale di click che si misura il guadagno dello spammer) devono essere accattivanti. Quelli con le immagini, specie di tipo pornografico, funzionano di più di quelli con semplice testo. Ma si tratta solo di una delle tante regole da seguire. Il lavoro dello Spammer è continuo, sette giorni su sette, non ci si può fermare perché altrimenti la concorrenza rischia di scavalcarvi. Una volta acquisita una "certa posizione" bisogna lottare con le unghie per mantenerla. Per il resto il lavoro dello Spammer ha anche diversi vantaggi. Innanzitutto l'orario di lavoro che, oltre a essere flessibile, non impegna più di qualche ora al giorno (5 di media) poi non occorre recarsi in ufficio, è praticabile ovunque.

L'IMPRENDITORE DELLO SPAM

L'attività di Spam può però non essere utilizzata solo per attività di phishing, ma semplicemente come veicolo commerciale per moltiplicare i guadagni su merce comprata all'ingrosso e rivenduta a 10 volte il suo prezzo d'acquisto. È il caso delle pillole farmaceutiche del tipo Cialis o Viagra che negli Stati Uniti vengono promosse attraverso l'attività di spam a circa 100 dollari e acquistate per 10 o 20 dollari in Cina o India. Ma diciamo che uno voglia specializzarsi semplicemente nell'invio di posta indesiderata per conto terzi, insomma fare lo spammer a pagamento. Quanto può guadagnare? I primi della classe, ovvero gli spammer più famosi, quelli che occupano, per intenderci, le prime posizioni di questa speciale classifica possono arrivare a mettere insieme anche 30.000 dollari al mese, anche se la media si aggira sui 6/7.000.
Chi pratica l'invio massiccio di posta per conto di terzi spesso non si preoccupa del contenuto. Egli veicola un messaggio e cerca di farlo nel modo più efficace possibile. In questa ottica l'attività dello spammer non è certo lecita, infatti, finisce per essere il vettore di un'attività il più delle volte fraudolenta, ma l'idea di non esercitare questa attività in forma diretta spesso rappresenta per lo spammer una sorta di attenuante personale, così che egli stesso non vive la sua attività come criminale ma come una qualsiasi altra attività imprenditoriale.


Da: Cole Boudreaux <carcinomas@royal-united.com>
Oggetto: Fw: important
Data: 14 febbraio 2011 11:41:25 GMT+01:00
A:
Rispondi a: Linkedin <support-26505098881476223875504872065575994058730860394901397017295054036314112767

V C L
I I E
G A V
R L I
A I T
S R
A

$1.29/pill $1.58/pill $2.81/pill

+ many other positions!

Visit Us at:
h t t p : / / v i a g r a 1 2 9 . c o m [delete spaces before visiting]

Viagra: un classico...

ARTIGIANI DEL CRIMINE E ORGANIZZAZIONI

La figura dello Spammer fin qui esaminata coincide, con tutte le differenze del caso, con quella del piccolo artigiano. Esistono tuttavia organizzazioni strutturate e ben più agguerrite che portano avanti questo genere di attività con volumi di traffico enormi e guadagni ben più superiori. In questo caso si parla di qualche centinaio di organizzazioni, non dissimili a quelle criminali che operano in altri ambiti, e che estendono la loro attività a livello globale. Il nocciolo della questione in fondo è sempre lo stesso: il denaro può spingere a fare qualsiasi cosa, e laddove ce n'è molto, attira spesso le persone più prive di scrupoli.



Finalmente in edicola la prima rivista
PER SCARICARE ULTRAVELOCE TUTTO quello che vuoi



WLF PUBLISHING

Chiedila subito al tuo edicolante!